Sigma: The Key for Vision-Language-Action Models toward Telepathic Alignment

Libo Wang
UCSI University
free.equality.anyone@gmail.com

Abstract

To address the gap in humanoid robot cognitive systems regarding the lack of a time-updable mediating thought space between semantics and continuous control, this study constructs and trains a VLA model named "Sigma" that runs on a single RTX 4090. It uses the open-source π0.5_base model as a foundation and preprocesses svla_so101_pickplace into a training dataset. The researcher independently designed an architecture for a vision-language-action model that combines deep semantic understanding and association to achieve telepathic communication. The training process involved repeated optimizations of data preprocessing, LoRA fine-tuning, and the inference-stage adapter. The experiment employed offline closed -loop replay, comparing Sigma with the untuned pure π0.5_base model under data conditions. Results showed that Sigma exhibited a stable decrease in control MSE across vector, fragment, and entire trajectory timescales, while maintaining the telepathy norm and semantic-text alignment quality unchanged. It demonstrates that mind-responsive alignment control is quantified through an architecture that combines deep understanding of semantics and association without retraining the base model, which provides reproducible experience for semantic alignment and intention-driven behavior in humanoid robots.

1. Introduction

With the development of vision-language-action (VLA) in humanoid robots, multiple parallel embodied technology routes have emerged instead of a single paradigm (Kawaharazuka et al., 2025). The successive releases of models such as RT-2 and OpenVLA have inspired more and more research teams to adopt a phased process that combines pre-trained VLM with fine-tuning of robot teaching data (Zitkovich et al., 2023; Kim et al., 2024). Current VLA models typically begin with network-scale representation learning using visual language backbones such as PaLI-X and PaLM-E; then, visual frames and language commands are integrated into chain-of-thought (CoT) reasoning to encode discrete latent tokens; finally, the quantized control sequence output by the action head is transferred through web knowledge to improve task generalization and semantic reasoning capabilities (Chen et al., 2023; Zhao et al., 2025; Shao et al., 2025). In contrast, while introducing a transformer-like backbone and multimodal tokenization, the Chinese team places greater emphasis on the whole-body dynamic coupling and contact stability with the specific organism (Vaswani et al., 2017; Zhong et al., 2025).

However, a crucial technological gap poses a gap for the VLA architecture: the lack of a continuously updated and interpretable mediating mental space between linguistic semantics and continuous control. This makes it difficult for the system to form a stable, structured reasoning chain when absorbing implicit context and inferring human intentions. This gap means that when instructions have multiple semantic layers, undefined operational goals, or require reliance on anthropomorphic associations to maintain action consistency, humanoid robots exhibit fragmented strategic decisions, intention deviations, and semantic misalignment problems (Sapkota et al., 2025). Meanwhile, both semantically driven transformer hubs and control-oriented routes emphasizing hardware coupling lack an abstraction layer capable of carrying high-level semantic context and precisely aligning with behavioral residuals due to this gap (Zhang et al., 2024). Consequently, when the VLA model fails to construct and maintain a potential thought space internally aligned with human cognitive structures, advanced tasks for humanoid robots often suffer from strategic imbalances and behavioral mismatches due to the disruption of "idea" transmission.

To address the gap between semantics and continuous control, which lacks a time-updable mediating cognitive space, this study trains and publishes a new VLA model named “Sigma” (Fig. 1). Telepathy is a VLA model that compresses the deep semantics and associative structures behind instructions into a

continuous internal thought state, and uses this state to align with unspoken human intentions to determine specific control behaviors. The researcher designs the telepathy factor τ as a latent cognitive vector shared across time, while simultaneously constructing modules for perception, reasoning, and behavior. At the vision level, visual basis tokens are generated through a multimodal encoder and perceiver-style resampling. These tokens are then refined by τ using FiLM-style gating and a transformer, thus constraining the scene representation within the current semantics and associative state.

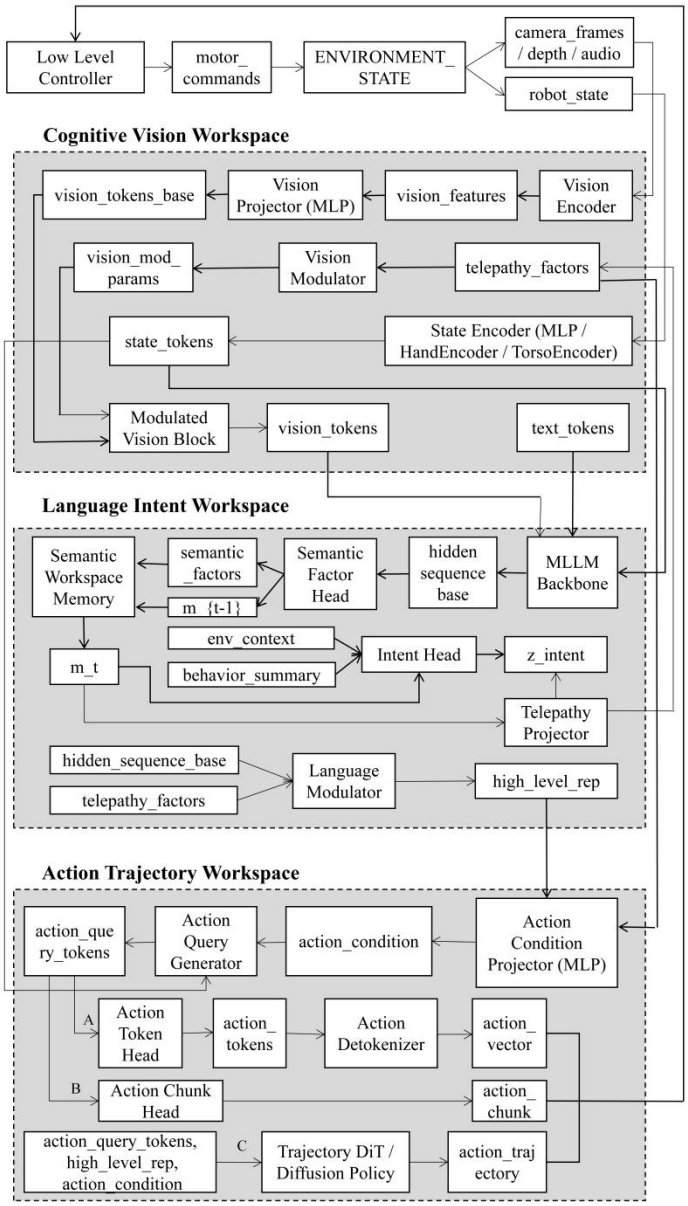

Fig. 1: Architecture of Sigma

On the language side, the MLLM backbone first extracts semantic factors from multimodal tokens, which are then aggregated into temporally continuous semantic memory via semantic workspace memory. It combines contextual information, behavioral summarization, and textual summarization to infer latent intentions, and generates τ by the telepathy projector, forming a shared thinking workspace capable of supporting "deep understanding of semantics and associations".

For the action module, Sigmafirst calculates the safe baseline behavior when τ is zero, and then maps τ and the higher-order representation to an explicit residual Δa through a telepathy residual head. After action

fusion and conversion into motor instructions by the low-order controller, this module achieves behavior modulation that approaches near-human mind-ready intention while preserving physical stability.

2. Related Work

Vision-Language-Action Models. From an architectural perspective, vision-language-action models are multimodal foundational strategies that simultaneously integrate vision, language, and robot actions (Din et al., 2024; Ma et al., 2024). They typically use a pre-trained visual language model as a semantic backbone, encoding camera observations and textual commands into latent representations, which are then directly output by an action decoder as joint trajectories or discrete control symbols that can be executed on the machine (Sapkota et al., 2025; Zhou et al., 2025; Zhang et al., 2025). A typical approach is to pre-train the encoder on network-scale image and text data, and then fine-tune it on a large dataset that matches images, sentences, and robot trajectories (Huang et al., 2025; Kim et al., 2025). The RT-2 is characterized by discretizing actions into text tokens and incorporating them into the language model vocabulary, while the OpenVLA and π0.5 series learn general control strategies from cross-vehicle data such as Open-X Embodiments (Zitkovich et al., 2023; Kim et al., 2024; O'Neill et al., 2024). More recent π0.5 and LeVERB further target multi-DOF humanoid and multi-arm systems, decoupling high-level semantic representation from low-level high-frequency control through a layered structure, which seeks a scalable trade-off between generalized semantic capabilities and fine-grained motion control (Black et al., 2025; Xue et al., 2025).

Multimodal Learning. Within a unified framework, multimodal learning is generally understood as processing heterogeneous signals such as language, vision, sound, and motion trajectories simultaneously, transforming previously separate perceptual channels into mutually interpretable representational structures through a shared latent space (Bouchey et al., 2021; Lu, 2023). This technique first designs dedicated encoders for each modality, compressing images, text, and state flows into aligned-dimensional vectors, and then forces the representations of the same event in different modalities to converge through contrastive learning or joint embedding loss (Bayoudh et al., 2022; Yu et al., 2024). Irrelevant samples are pushed away, thus forming stable cross-modal correspondences in the latent space. With the popularization of transformer architecture, cross-modal attention and multi-layer interactive coding have gradually become mainstream (Vaswani et al., 2017; Xu et al., 2023; Yuan et al., 2025). This prompts the model to simultaneously pay attention to text fragments and image regions in the same layer of computation and repeatedly update the relationship between them in multiple iterations, thereby further supporting downstream alignment, retrieval, and decision control (Tang et al., 2022; Zhang et al., 2023).

3. Architecture

Unlike the thought communication proposed by Zheng et al. (2025) that breaks through the limitations of natural language, the architecture transforms latent thinking from multi-agent communication to a shared space of telepathy factors and semantic memory within a single humanoid robot. It aims to directly modulate baseline control strategies with residuals, concretely realizing mind-reading as the intrinsic alignment between perception and action.

3.1 Cognitive Vision Workspace

The module starts with environmental feedback. Motor commands generated by the low-level controller change the environment state. The sensor outputs camera_frames, depth, audio, and robot_state, which are sent to the vision encoder and state encoder, respectively. Internally, PatchEmbed Conv and AudioPatchEmbed convert multi-source signals into variable-length tokens, which are then concatenated into vision_features. The vision projector projects these features uniformly onto the *d*_model space of the MLLM using two layers of MLP. Subsequently, a perceiver-style resampler constructs a fixed-length vision_tokens_base using learnable queries to ensure that the downstream language module faces a stable N_v vision base.

The state encoder compresses the robot_state using MLP and expands it into N_s state_tokens via token_expander, aligning proprioception and vision within the same representation space. Telepathy_factors are adjusted using a learnable scale $\tau_{\text{log\_scale}}$ and fed into the vision modulator to generate the FiLM parameters gamma and beta. It performs channel-wise scaling and translation on the vision_tokens_base, and then uses mod_log_scale as a post-valve value to achieve continuous interpolation from no psychic signals to emphasizing latent thoughts. Parallel text commands are converted to text_tokens externally by the tokenizer. Although not rewritten within this module, they share the *d*_model

space with vision_tokens and state_tokens, allowing them to participate in cross-modal reasoning alongside telepathy factors in the subsequent language module.

The modulated vision block refines local relationships on v_mod using multi-layered self-attention and feedforward networks, outputting semantically saturated and telepathy-modulated vision_tokens. These, along with state_tokens, are sent to the language intent workspace module to form the perceptual entry point for the entire mind-sensing chain. The corresponding algorithms are shown below:

$$F \in \Re^{N_f \times d},\ Q \in \Re^{N_f \times d},\ \tau \in \Re^{d_\tau}$$

$$A = \mathrm{softmax}(\frac{QW_Q(FW_K)^{\mathrm{T}}}{\sqrt{d}}),\ H = A(FW_V)$$

$$V_{\mathrm{base}} = \mathrm{FFN}\,(\mathrm{LN}(\mathrm{Q}+\mathrm{H})),\ V_{\mathrm{base}} \in \Re^{N_v \times d}$$

$$\tau' = e^{\theta_\tau}\tau,\ h = \sigma(\tau' W_1 + b_1),\ [\gamma, \beta] = hW_2 + b_2$$

$$V_{\mathrm{mod}} = V_{\mathrm{base}} + e^{\theta_{\mathrm{mod}}}(V_{\mathrm{film}} - V_{\mathrm{base}}),\ V_{\mathrm{mod}} \in \Re^{N_v \times d}$$

where F is the multimodal visual feature sequence output by the vision encoder and projector, N_f is the number of visual tokens in the sequence; Q is the query token sequence used for resampling by the perceiver; N_v is the fixed length of the output visual tokens; d is the shared feature dimension dmodel; W_Q, W_K, and W_V are the linear projection matrices of query, key, and value, respectively; A is the attention weight matrix calculated; H is the feature vector aggregated according to A; LN represents the LayerNorm applied to the token sequence; FFN represents the feedforward sublayer;Vbase is the fixed-length visual base token sequence obtained after resampling; τ represents the telepathy factor vector from the language module; d_τ is its dimension; θ_τ is the log-scale gate parameter controlling the telepathy intensity; and τ' is the factor amplified by θ_τ. W_1, W_2 and b_1, b_2 are the weights and biases of the MLP inside the vision modulator; $\sigma(\cdot)$ is the GELU nonlinearity used therein; γ and β are the FiLM scaling and translation coefficients generated by τ'; V_{base} is the unmodulated visual base token; V_{film} is the intermediate visual representation after applying FiLM; θ_{mod} is the second log-scale gate parameter controlling the modulation amplitude; and V_{mod} is the final output telepathy modulated visual token.

3.2 Language Intent Workspace

After constructing a high-level linguistic thought field that spans time, semantics, and context, all inputs to this module are concatenated using text tokens, vision tokens, and state tokens. It uses an MLLM backbone to form a hidden sequence base through multi-layered cross-modal attention, aligning language, vision, and ontological state within the same d_{model} space. Semantic factors are read from the hidden sequence base using a learnable query by the semantic factor head, generating K semantic factors. At each time step, the semantic workspace memory first reads the semantic memory vector $m_{t\text{-}1}$ from the previous time step, then integrates the current semantic factors into mt according to a gating recursive formula to preserve semantic continuity across time.

Three summary heads extract env_context, behavior_summary, and text_summary from the hidden_sequence_base with independent parameters, enabling the model to separate environmental clues, behavioral trends, and linguistic context. The Intent head merges m_t, c_{env}, and c_{beh} to infer z_{intent}, which serves as the semantic driver for telepathy, and the telepathy projector combines m_t, z_{intent}, c_{env}, c_{beh}, $z_{\mathrm{sem\_pool}}$, and c_{text} to generate telepathy_factors as the global semantic alignment vector. The language modulator then applies a double gating bias modulation to the hidden_sequence_base with τ_t, causing the language representation to converge toward the inferred intrinsic intent, and finally outputs the high-level semantic representation high_level_rep, which feeds back to other modules to form a cross-modal closed loop.The specific algorithm is as follows:

$$Z_{\mathrm{sem}} \in \Re^{K \times d},\ Z_{\mathrm{pool}} = \frac{1}{K}\sum_{i=1}^{K} Z_{\mathrm{sem},i}$$

$$u = \mathrm{GELU}\ (W_u z_{\mathrm{pool}} + b_u),\ \lambda = \sigma(W_\lambda [m_{t\text{-}1}; z_{\mathrm{pool}}] + b_\lambda),\ m_t = \lambda \odot m_{t\text{-}1} + (1 - \lambda) \odot u$$

$$x = [m_t; z_{\mathrm{intent}}; c_{\mathrm{env}}; c_{\mathrm{beh}}; z_{\mathrm{sem\_pool}}; c_{\mathrm{text}}]$$

$$h = \mathrm{GELU}(W_1 x + b_1),\ \tau_t = W_2 h + b_2$$

Where $m_{t\text{-}1}$ represents the semantic memory vector at the previous time step in semantic memory propagation; z_{sem} is the semantic factor matrix read from the semantic factor head at the current time step; z_{pool} is the average pooling result of z_{sem}; u is the candidate semantic signal after updated projection; λ is the gating coefficient for interpolation between the old memory $m_{t\text{-}1}$ and the candidate update u; m_t is the current semantic memory vector obtained after integration; in the telepathy projector, mt serves as the semantic memory summary; z_{intent} is the latent intent vector inferred from the intent head; c_{env} and c_{beh} correspond to contextual summarization and behavioral trend summarization, respectively; zsem_pool is the pooling representation of semantic factors at each time step; c_{text} is the text summary. These six vectors are concatenated to form a fusion vector x, h represents the implicit semantics of x after transformation by a multi-layer perceptron; and τ_t represents the final projected telepathy_factors used to modulate higher-order semantic alignment in other modules.

3.3 Action Trajectory Workspace

This module is responsible for converting the language module's high_level_rep and the current perception state into executable control trajectories. The action condition projector receives the high_level_rep and telepathy_factors, and generates action_conditions through an MLP as condition vectors for subsequent planning stages. The action query generator combines the action_conditions with state_tokens, performs cross-attention and multi-layer transformer refinement on the learnable query seeds, and outputs action_query_tokens as the common query basis for the three action branches.

Along path A, the action token head compresses action query tokens into low-dimensional action tokens, which are then reduced by the action tokenizer to a continuous action vector for single-step or high-frequency joint control. Along path B, the action chunk head constructs action chunks using pooling and linear projection to characterize short-duration, consistent action segments within a single slice. Along path C, the trajectory DiT/diffusion policy uses action query tokens, high-level reps, and action conditions to progressively denoise the noisy trajectory, generating an action trajectory that corresponds to a longer-term motion plan. Ultimately, action_vector, action_chunk, and action_trajectory are reweighted and decoded into motor_commands in the action fusion and low-level controller, closing the control loop from high-level intent to actual joint drive. The relevant algorithms are as follows:

$$Q_{\text{l}} = \text{Attn}(Q_0, S_{\text{proj}}, S_{\text{proj}}) + b(c_{\text{act}}),\ q_t = \text{LN}(\text{Ref}(Q_{\text{l}}))$$

$$c_{\text{act}}^{\text{base}} = P_{\text{act}}(r_{\text{high}}, 0),\ q_{\text{base}} = Q(c_{\text{act}}^{\text{base}}, S_t)$$

$$a_{\text{vec}}^{\text{base}} = H_A(q_{\text{base}}),\ a_{\text{chunk}}^{\text{base}} = H_B(q_{\text{base}}),\ a_{\text{traj}}^{\text{base}} = D_{traj}(q_{\text{base}}, r_{\text{high}}, c_{\text{act}}^{\text{base}})$$

$$[\Delta a_{\text{vec}}, \Delta a_{\text{chunk}}, \Delta a_{\text{traj}}] = \text{g}([r_{\text{high}}, \tau])$$

$$a_{\text{vec}}^{\tau} = a_{\text{vec}}^{\text{base}} + \Delta a_{\text{vec}},\ a_{\text{chunk}}^{\tau} = a_{\text{chunk}}^{\text{base}} + \Delta a_{\text{chunk}},\ a_{\text{traj}}^{\tau} = a_{\text{traj}}^{\text{base}} + \Delta a_{\text{traj}}$$

$$u_t = \phi(a_{\text{vec}}^{\tau}, a_{\text{chunk}}^{\tau}, a_{\text{traj}}^{\tau}),\ m_t = C_{\text{low}}(u_t)$$

where S_t represents state_tokens; W_s is the linear projection matrix that aligns them to d_model; Q_{seed} is the learnable query template; Attn(·) represents multi-head cross-attention operation; c_{act} is the condition vector generated by the action condition projector; b(·) is the bias it applies to the query; Ref (·) is the transformer encoder on the query; LN(·) is layer normalization; q_t is action_query_tokens; P_{act} represents the action condition projector; Q represents the action query generator; H_A and H_B correspond to the action token head and action chunk head, respectively; D_{traj} corresponds to the trajectory DiT/diffusion policy; $a^{\text{base}}{}_{\text{vec}}$, $a^{\text{base}}{}_{\text{chunk}}$, and $a^{\text{base}}{}_{\text{traj}}$ are the three-way baseline actions; g(·) is the MLP of the telepathy residual head; Δa_{vec}, Δa_{chunk}, and Δa_{traj} are the telepathy residuals, $a^{\tau}{}_{\text{vec}}$, $a^{\tau}{}_{\text{chunk}}$ and $a^{\tau}{}_{\text{traj}}$ are the final action branches; ϕ(·) is action fusion; u_t is the control representation after fusion; C_{low} is the low-level controller; m_t corresponds to the motor_commands output after flowing to the low-level controller in the graph.

4. Train

This study employs a complete and transparent implementation trajectory for its training process. The researcher first preprocesses multimodal sequence data using the PyTorch framework, ensuring that visual, linguistic, and proprioceptive signals are absorbed by the model within a unified structure. Subsequently, LoRA fine-tuning was performed on the current open-source VLA model π0.5_base using a single

RTX4090 as the basis, that is, to enhance the stability of semantic-action mapping in a low-rank adaptation manner. The performance of the computing device is shown in Fig. 2.

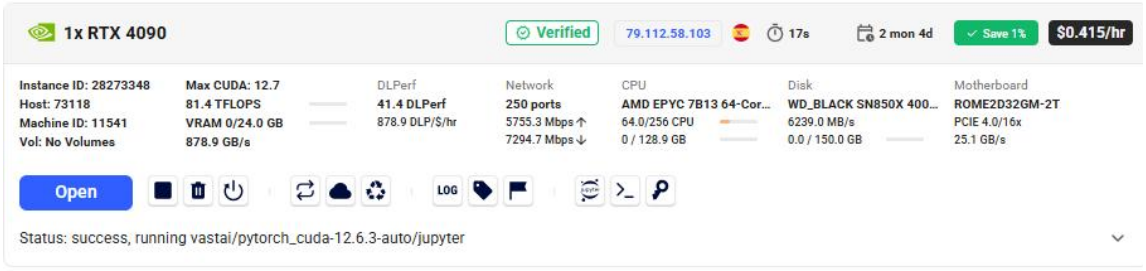

Fig. 2: Detailed computing power configuration of a single NVIDIA RTX 4090

In addition, the researcher combined interventional adapter technology to give the model controllability and more nuanced semantic correction capabilities in key inference areas. All preprocessed data, fine-tuned parameters, training logs, and inference results have been fully disclosed in the Hugging Face repository to ensure the research's verifiability and reproducibility.

4.1 Data Preprocessing

The preprocessing stage is automated throughout the entire process using scripts. First, load_sigma_env reads the environment variables and HF_token from sigma.env. Then, load_lerobot_dataset loads trajectory data from svla_so101_pickplace. Subsequently, prefetch_hf_dataset, along with exponential backoff and 429 detection, is used solely for warming up the Hugging Face cache, improving the stability of large-scale downloads. Data iteration is handled by safe_iter_dataset, providing multiple retries and 429 backoff mechanisms for each step. Simultaneously, samples are aggregated into episodes based on episode_index, and sorted by frame_index when necessary to restore temporal order.

For each episode, the researcher used a sliding window with horizon_T=16 to divide the trajectory using build_windows. Within each window, pick_rgb_from_example, pick_state_from_example, pick_action_from_example, and pick_text_from_example are called to extract the corresponding screen sequence, robot state, action sequence, and text command, which are then further calculated using compute_action_stats to calculate the average and maximum L2 norm. These were then further processed using compute_action_stats to calculate the average and maximum L2 norm. If the average norm was lower than the min_action_norm threshold, it was considered a nearly static segment and directly filtered to ensure that training focused only on windows with actual operations. For each retained window sample, the script constructed a standardized sample dictionary. It synchronously maps the original fields such as vision, state, action, and text to statistics such as vision_inputs, robot_state, gt_action_vector, gt_action_chunk, gt_action_trajectory, avg_action_l2, and max_action_l2 used in subsequent training.

The sample writing is managed by ShardWriter, which checks existing shard_*.pt files during initialization to determine the starting shard number and skip_count to avoid duplicate writes. It outputs shard_00000.pt, shard_00001.pt, shard_00002.pt, etc., sequentially after a fixed shard_size buffer. Simultaneously, the generated meta.json records detailed data set IDs, episode counts, window statistics, and all preprocessing hyperparameters to provide a traceable engineering foundation for subsequent fine-tuning and evaluation.

4.2 LoRA Fine-Tuning

As a LoRA fine-tuning phase of the Sigma, the researcher implemented the entire training process using a complete engineering pipeline. The training process first loads a visual language base (Black et al., 2025) represented by π0.5_base from the Hugging Face or locally. It combines r=16 with a LoRA structure with dropout, only unfreezing the q, k, v, and output projection layers to reduce the number of parameters, while supporting four-bit quantization and accelerated mixed precision and gradient accumulation to control computational overhead. Subsequently, the Sigma shard dataset is constructed using the aforementioned shard file. A custom collator encodes the text commands into one-hot encoding and then aligns them to d_{model} via fixed linear projection. Furthermore, the process unifies the multi-frame visual features and the compressed robot_state into vision_inputs and state tensors, and simultaneously derives vector-level, chunk -level, and trajectory-level ground truth actions, as well as optional baseline actions, as a foundation for subsequent residual learning.

This VLA model uses the architecture proposed above to combine the three sub-modules of vision, language, and action into a single forward path. At each step, a visual forward pass without telepathy is executed first, and then the language module generates telepathy_factors and high_level_rep, which are fed back to the vision and action modules to produce three action outputs. The original algorithms in this process had a significant impact on the fine-tuning process as follows.

4.2.1 Telepathic Residual Action Focusing

Telepathic Residual Action Focusing (TRAF) algorithm uses a baseline action of $\pi 0.5$ as a reference and learns only the telepathic residuals of the three action outputs to map high-level semantics into fine-tuning of action_vector, action_chunk, and action_trajectory. Simultaneously, it calculates sample-level errors and extracts top-k difficult segments with additional weights, allowing the model to focus its learning capacity on the most challenging humanoid robot alignment scenarios while maintaining stable control.

For each batch, the model outputs three raw actions:

$$\Delta a_{\text{vec}}, \Delta a_{\text{chk}}, \Delta a_{\text{traj}}$$

If the data also provides the baseline ($\pi 0.5$) action abase, then the final action is:

$$a_{\text{vec}} = a_{\text{vec}}^{\text{base}} + \Delta a_{\text{vec}},\ a_{\text{chk}} = a_{\text{chk}}^{\text{base}} + \Delta a_{\text{chk}},\ a_{\text{traj}} = a_{\text{traj}}^{\text{base}} + \Delta a_{\text{traj}}$$

Otherwise, $a. = \Delta a.$, the corresponding annotation is:

$$a_{\text{vec}}^{*}, a_{\text{chk}}^{*}, a_{\text{traj}}^{*}$$

The global action loss is

$$L_{\text{act}} = \alpha_a \text{MSE}(a_{\text{vec}}, a_{\text{vec}}^{*}) + \alpha_b \text{MSE}(a_{\text{chk}}, a_{\text{chk}}^{*}) + \alpha_{\text{c}} \text{MSE}(a_{\text{traj}}, a_{\text{traj}}^{*})$$

Calculate the difficulty of each sample based on the telepathy-corrected final action:

$$h_i = \text{MSE}_i(a_{\text{vec}}, a_{\text{vec}}^{*}) + \text{MSE}_i(a_{\text{chk}}, a_{\text{chk}}^{*}) + \text{MSE}_i(a_{\text{traj}}, a_{\text{traj}}^{*})$$

Select the top-k subsets (with a proportion of ρ) to form set H, and then calculate a difficult sample loss on this subset:

$$L_{\text{hard}}^{\text{act}} = \alpha_a \text{MSE}_H(a_{\text{vec}}, a_{\text{vec}}^{*}) + \alpha_b \text{MSE}_H(a_{\text{chk}}, a_{\text{chk}}^{*}) + \alpha_c \text{MSE}_H(a_{\text{traj}}, a_{\text{traj}}^{*})$$

The final action item is

$$L_{\text{act,total}} = L_{\text{act}} + \lambda_{\text{hard}} L_{\text{act}}^{\text{hard}}$$

where a_{vec}, a_{chk}, and a_{traj} are the final vector-level, fragment-level, and trajectory-level action outputs; a_{chk} is the corresponding label; $\Delta a.$ is the telepathy residual head prediction; $a_{.}^{\text{base}}$ is the offline $\pi 0.5$ baseline action; α_a, α_b, and α_c are the loss weights of the three actions; h_i is the difficulty score of the i-th sample; p is the proportion of difficult samples; H is the set of top-k difficult samples; λ_{hard} is the loss weight of difficult samples.

4.2.2 Telepathic Semantic Alignment Curriculum

Telepathic Semantic Alignment Curriculum (TSAC) progressively enhances the alignment weights between semantic memory, intention vectors, and telepathy factors during the training process. It first stabilizes basic control through action regression, and then linearly increases semantic consistency and directional regularity with each step, which forces the model to converge its internal thought structure and explicit trajectory to the same mental coordinate system in the later stages. The original algorithm is as follows:

Semantic consistency loss is composed of the semantic factor pooling vector z_{pool}, the previous moment's memory m_{prev}, and the text/vision pooling vector.

$$L_{\text{sem}} = L_{\text{time}}(z_{\text{pool}}, m_{\text{prev}}) + \beta_{\text{mi}} L_{\text{mi}}(z_{\text{pool}}, \text{text} + \text{vision})$$

Intent association loss involves aligning the intent vector z_{intent} with the current semantic memory m_t in terms of direction:

Telepathy regular expressions simultaneously control the norm of τ and align it with the direction of the action condition c_{act}:

$$L_\tau = 0.01 \| \tau \|_2^2 + \lambda_{\text{collapse}} [\max(0, \tau_0 - \| \tau \|_2)]^2 + \eta_{\text{var}} (1 - \cos(\tau, c_{\text{act}}))$$

Overall loss per training step t using course weighting:

$$w_{\text{sem}}(t), w_{\text{intent}}(t), w_\tau(t)$$

Linear interpolation from the initial value to the target value yields:

$$L_{\text{total}}(t) = L_{\text{act,total}} + w_{\text{sem}}(t) L_{\text{sem}} + w_{\text{intent}}(t) L_{\text{intent}} + w_\tau(t) L_\tau$$

where z_{pool} is the semantic factor pooling vector; mprev and mt are the semantic memories of the previous and current moments, respectively; L_{time} measures the consistency of semantic memory across time; L_{mi} is the mutual information comparison loss between the semantic vector and the text/vision fusion representation; β_{mi} is its weight; z_{intent} is the intention vector; τ is the telepathy_factors; c_{act} is the action_condition; $\lambda_{\text{collapse}}$ controls the strength of the anti-collapse term; τ_0 is the minimum norm threshold of the expectation; η_{var} is the direction alignment regularization weight; $w_{\text{sem}}(t)$, $w_{\text{intent}}(t)$, and $w_\tau(t)$ are the course weights that increase linearly with the number of training steps, making the early stage mainly action regression; the late stage then gradually strengthens semantic and telepathy alignment.

4.3 Adapter

As an auxiliary scheme to optimize LoRA performance, an adapter can be constructed to intervene in Sigma's behavioral output in a hook-and-loop manner after the inference phase, rather than modifying the underlying weights (He et al., 2021). It simultaneously reads the model-predicted action_vector, chunk, trajectory and the offline base_action of π0.5_base, and treats the difference between the two as the telepathy residual Δa_τ. It uses the residual ratio of Δa_τ relative to the baseline norm, the L2 norm of telepathy_factors, and the cosine similarity between telepathy_factors and action_condition to form a risk score, and obtains a continuous scaling factor between minscale and maxscale through exponential mapping. When the residual magnitude is moderate and τ aligns with the action conditions, the adapter strengthens the telepathy residual weights, allowing the model to fully utilize high-level semantic corrections. If the residual becomes abnormal or τ deviates from the target range, the gate automatically converges to near zero, reverting the output to the π0.5 baseline behavior. Finally, the adapted action_vector, chunk, trajectory, along with scaling factors and risk metrics, are output together, preserving the intuitive adjustments provided by telepathy while employing fine-grained risk control to avoid disrupting the original stable control.

4.4 Loss Dynamics

Given the established LoRA and adapter engineering pipeline, the researcher examined the loss dynamics of the training process. Table 1 shows that, based on multiple monitoring points from epoch 0, the total loss is almost entirely dominated by L_{act}, while L_{sem} remains stable at around 0.07. L_{int} gradually transitions from near zero to a significantly negative value, indicating that the cosine similarity between the intention vector and semantic memory gradually increases after the course weights are increased, thus preventing semantic collapse. As w_{sem}, w_{int}, and w_{tau} linearly increase, τ_{rms} smoothly rises from around 0.04 to approximately 5.6, indicating that the telepathy factor norm is gradually opened up within a controllable range. Meanwhile, L_τ decreases from 0.17 to approximately 0.06–0.33. Coupled with stable hard_ratio=0.30 and fluctuations of the same magnitude in $L_{\text{act\_hard}}$, the overall gradient explosion or mode collapse is not triggered.

Table 1 summarizes the statistics of these indicators at different step sizes, showing that action regression, semantic alignment, and telepathy regularization maintain a balance in magnitude. For Sigma, this means that without sacrificing baseline control stability, the training successfully transferred a portion of the representational capacity to deep "semantic-intention-action" alignment, which provides a quantifiable advantage for the telepathic effect in subsequent experiments.

Table 1: Decomposed training loss and telepathy activation profile

step/gstep	loss	L_act	L_sem	L_int	L_tau	L_act_hard	w_sem	w_int	w_tau	hard_ratio	tau_rms
0	1086.908	1086.898	0.069	0.032	0.172	618.710	0.100	0.100	0.000	0.300	0.049
10	1914.058	1914.036	0.069	0.035	0.171	1141.169	0.210	0.185	0.004	0.300	0.049
20	1308.330	1308.308	0.069	-0.004	0.169	716.477	0.320	0.271	0.007	0.300	0.044
30	1033.426	1033.469	0.069	-0.209	0.156	573.119	0.429	0.356	0.011	0.300	0.054
40	1838.295	1838.500	0.073	-0.558	0.112	1100.001	0.539	0.441	0.015	0.300	0.146
50	1103.291	1103.584	0.069	-0.645	0.090	580.732	0.649	0.527	0.018	0.300	0.241
60	1426.918	1427.275	0.069	-0.671	0.072	748.638	0.759	0.612	0.022	0.300	0.381
70	2036.956	2037.376	0.069	-0.691	0.058	1181.513	0.868	0.698	0.026	0.300	0.566
80	2080.200	2080.660	0.118	-0.737	0.045	1148.737	0.978	0.783	0.029	0.300	1.075
90	2319.583	2320.092	0.069	-0.725	0.049	1162.648	1.000	0.800	0.030	0.300	1.435
100	1975.697	1976.183	0.069	-0.696	0.061	1133.261	1.000	0.800	0.030	0.300	1.898
110	1112.991	1113.447	0.069	-0.659	0.085	585.033	1.000	0.800	0.030	0.300	2.503
120	1058.931	1058.888	0.525	-0.610	0.202	535.251	1.000	0.800	0.030	0.300	4.292
130	1669.481	1669.870	0.069	-0.586	0.337	835.426	1.000	0.800	0.030	0.300	5.653

5. Experiments

The researcher conducted offline closed-loop replays comparing the control behavior of Sigma and π0.5_base on the same dataset. The experiments were based on preprocessed shard-generated trajectories, and the success rate and trajectory error were statistically analyzed with the telepathy switch enabled and disabled to comprehensively assess the significance of Sigma's telepathic abilities. Part of the experimental design scripts, tools, and logs are publicly available on Hugging Face.

5.1 Setup

In view of a fixed pipeline, the experiment can be set up with the presence or absence of a telepathy layer within the same backbone as the core control. The experimental group uses a Sigma model based on π0.5_base for LoRA fine-tuning and adaptation by loading sigma_telepathy_heads.pt from the Hugging Face repository. The researcher enabled the "Telepathy" switch, allowing high_level_rep and telepathy_factors to actually participate in the control decision. The control group directly calls the original pi0.5_base model without loading any fine-tuning weights and adaptation scripts, meaning that the output of this group is equivalent to the open-source baseline behavior. Both VLA models were replayed in a closed loop on identical visual and state sequences, and task-level metrics such as success rate and trajectory deviation were compared.

This design avoids interference from environmental randomness and data variations on the results, strictly locking the variation to the single factor of whether telepathy is switched on or off. It provides strong evidence for evaluating whether Sigma truly achieves a control advantage of "deep semantic understanding + association → telepathy" in humanoid robot scenarios.

5.2 Dataset

As mentioned earlier in the training phase, the experimental dataset for this study comes from the open-source manipulation dataset svla_so101 pickplace on Hugging Face. After data preprocessing and reorganization using a sliding window with horizon_T=16, three publicly available shard files, shard_00000.pt, shard_00001.pt, and shard_00002.pt, were finally exported as the sole source of the experiments. Each shard contains aligned visual sequences, robot states, and continuous motion trajectories. At the same time, the distribution is concentrated on real-world manipulation segments by filtering out nearly static windows using a minimum motion norm threshold. It means that a single upstream dataset combined with a strictly traceable preprocessing design avoids the dilution effect of task and scene noise, and allows subsequent comparisons of the differences between Sigma and π0.5_base to be more directly interpreted as differences in representation and control brought about by the mind-sensing layer.

5.3 Implementation

The experiment focused on offline closed-loop replay. The researcher first loaded the environmental variables and HF_token, and automatically downloaded sigma_pickplace and sigma_telepathy_heads.pt when necessary via ensure_sigma_artifacts. It used the LeRobot's π0.5_base policy as the control backbone,

simultaneously extracting aligned tokenizers and text embedding layers and checking vocabulary consistency. Next, it constructed the Sigma shard dataset and data loader based on the shard directory. Each batch maintained aligned vis_obs, robot_state, texts, three ground truth actions, and optional base_action_*. Simultaneously, this process used the internal embedding of π0.5_base to generate text_tokens, corrected the robot_state dimension, and then fed them in. The main loop repeatedly forwards under the conditions of enabling and disabling "telepathy" and whether to use an adapter. On one hand, it records the branch MSE of action_vector, chunk, and trajectory, the L2 of telepathy_factors, and the cosine alignment of semantic_factors with respect to the text. On the other hand, it accumulates the proportion of difficult samples and the average error based on the success threshold and hard threshold, and finally summarizes them into sigma_eval_report.json, which is published on Hugging Face along with the batch log.

6. Result & Discussion

In the data results, both groups completed the evaluation under the conditions of num_samples=723 and num_batches=181. The differences are mainly reflected in three MSE indicators to measure the mean squared error of the stepwise action_vector relative to the true value: avg_mse_vector is approximately 79.03 (Sigma) and 98.83 (π0.5_base), respectively; avg_mse_chunk is approximately 203.05 vs. 228.97, corresponding to the reconstruction error of short-time fragment-level action_chunks; and avg_mse_traj is approximately 174.71 vs. 191.03, reflecting the bias of long-time domain action_trajectory. The values of avg_tau_l2=51.60 and avg_semantic_text_alignment=0.1307 are completely consistent in both groups, proving that the telepathy factor norm and semantic-text alignment strength themselves do not change due to different conditions, but rather the final behavioral quality is determined by whether the model effectively utilizes these signals. Since hard_thresholds is fixed at vec=0.1, chk=0.2, and trj=0.2, and hard_sample_fraction=1.0 and total_hard_samples=723, avg_hard_mse_vector / chunk / traj are almost identical to all samples. This can be seen as providing direct quantitative evidence that Sigma consistently outperforms the untuned π0.5_base at all three time scales (vector, fragment, and trajectory) in terms of control precision improvement brought by the mind-sensing layer, under the assumption that all windows are treated as difficult samples.

In contrast, CHECK A was specifically used to check whether the structure and scale of the telepathy weights themselves remained completely consistent between the experimental and control groups. Both groups read sigma_telepathy_heads.pt, but the control group did not use the telepathy factor in its control strategy. The statistics were: heads_tensors=325, indicating that there were 325 independent tensors forming the telepathy heads; mean=0.002, indicating that the overall average of all weight values was close to zero, avoiding global bias; std=0.107, reflecting that the standard deviation of the weight distribution was approximately 0.107; and rms=0.107, indicating that the weight energy in the squared average sense was also on the same order of magnitude.Since the four values mentioned above were exactly the same in the experimental and control groups, it was confirmed that the subsequent behavioral differences did not come from differences in the telepathy weight file or the initial geometry, but rather from whether or not this set of telepathic representations was enabled and actually used in control decisions.

In addition, CHECK B uses multiple behavioral and representational metrics to characterize the actual differences between the two models. In table 2 and table 3, mse_vec is defined as the mean squared error of the stepwise action vector relative to the true value of the trajectory, used to measure the accuracy of fine-grained control at a single time step. mse_chk calculates the MSE over a fixed-length action segment to reflect the stability of the action in direction and amplitude over a short period. mse_trj treats the entire action trajectory as a high-dimensional curve, and its MSE corresponds to whether the long-term planning can fit the ideal trajectory overall. tau_l2 observes the L2 norm of telepathy_factors, which is an indicator of the strength at which the telepathy channel is actually opened. It is used to check whether Sigma maintains adequate representational energy when behaving well. sem_align measures the alignment between semantic factors and text embeddings. Its role is to verify whether semantic-text-behavior is compressed into a consistent mental coordinate system, which provides semantic evidence for subsequently interpreting behavioral differences as effects of the telepathy layer.

Table 2: The metrics of experimental group - CHECK B

Model	batch	mse_vec	mse_chk	mse_trj	tau_l2	sem_align
Sigma	0	61.835	292.177	251.009	51.593	0.131
	20	113.477	182.101	159.574	51.599	0.131
	40	49.340	236.021	211.508	51.598	0.131
	60	50.503	214.079	187.492	51.599	0.131
	80	108.293	168.418	150.344	51.600	0.131
	100	45.875	208.893	188.591	51.600	0.131
	120	71.466	299.924	250.684	51.594	0.131
	140	149.410	246.790	207.350	51.591	0.131
	160	69.113	293.315	253.594	51.593	0.131
	180	33.893	163.460	149.573	51.603	0.131

Table 3: The metrics of control group - CHECK B

Model	batch	mse_vec	mse_chk	mse_trj	tau_l2	sem_align
π0.5_base	0	120.382	329.023	273.857	51.593	0.131
	20	118.060	199.856	170.461	51.599	0.131
	40	104.931	267.907	232.370	51.598	0.131
	60	88.294	240.883	204.772	51.599	0.131
	80	111.947	184.658	160.407	51.600	0.131
	100	92.778	237.755	207.414	51.600	0.131
	120	124.780	337.781	274.761	51.594	0.131
	140	166.116	275.206	227.315	51.591	0.131
	160	130.318	330.571	277.000	51.593	0.131
	180	64.938	188.277	165.839	51.603	0.131

Tables show that, with identical num_batches and hard thresholds, Sigma's control errors across the three time scales are generally lower than π0.5_base. Statistically, mse_vec decreases by approximately 20%, mse_chk by approximately 10%, and mse_trj by nearly 10%. In batches 0 and 180, representing the initial and final states, Sigma's vector and trajectory errors are mostly close to half of π0.5_base. It is worth noting that tau_l2 and sem_align are almost identical across batches, indicating that the energy scale of the telepathy factor and the semantic alignment quality itself have not changed. Only when these features are actually translated into action corrections by the TRAF and TSAC pipelines in Sigma does it lead to a reduction in coherence error across single steps, fragments, and the entire trajectory. From the perspective of the evidence chain, this phenomenon of obtaining stable control gain solely due to semantically driven residuals under the same semantic and telepathy geometry indicates that the Sigma model has, to some extent, internalized deep semantics and associations into observable telepathic control advantages, but there is still room for improvement before it can be fully aligned with human understanding.

7. Limitation & Future Research

Although this study validated the advantages of the Sigma model in telepathic control in a single pick-place scenario, several boundary conditions still need to be strengthened. All current analyses are based on the svla_so101_pickplace and π0.5_base backbone; the semantic factors and telepathy representations have not yet been stress-tested across more mission families, more machine types, and multi-turn dialogue commands, leading to conservative generalizations regarding cross-mission and cross-machine types. In addition, while the offline replay metrics employed by the researchers can characterize the quality of control after semantic alignment, they have not yet been combined with subjective human assessments or long-term deployment of real humanoid robots to examine the robustness of telepathic links under noise perception, contact uncertainty, and safety constraints. Future research needs to expand to include multi-task, multi-modal datasets and different VLA backbones to observe the stability of telepathy in higher-dimensional behavioral spaces. Furthermore, VLA implementation needs to combine online fine-tuning with experiments on physical humanoid robots to further solidify the deep semantic understanding and associations currently observed indirectly through metrics into alignment with human telepathy.

8. Conclusion

Based on the open-source π0.5_base, this study constructs the vision-language-action model "Sigma" through self-designed architecture fine-tuning and training. It expands the original baseline of simply mapping perception-thought-action to a three-layered workspace that simultaneously models semantic memory, intention vectors, and telepathy factors. The vision module provides an intervened perceptual basis through modulated vision/state tokens; the language module maintains mt and zintent in the semantic workspace; and the action module outputs vectors, fragments, and trajectory control through three residual branches. The Sigma model as a whole uses LoRA fine-tuning and an intervention-response inference adapter to form a reproducible training and deployment process. The experiment used a single RTX 4090 to save computing power costs, importing backbone weights, sigma_telepathy_heads.pt, and pick-place data shards. The data from CHECK A confirm that the telepathy and semantic alignment conditions are completely consistent in the experimental and control groups; CHECK B shows that, under the premise that tau_l2 and sem_align are equal, Sigma's control errors on the three time scales of mse_vec, mse_chk, and mse_trj are all consistently lower than the π0.5_base baseline. In summary, this study presents a feasible path to transform existing VLA models into semantic-intention-action alignment capabilities without retraining the backbone. It provides preliminary quantitative evidence demonstrating the feasibility of combining deep semantic understanding and association to achieve telepathic communication in humanoid robots and offers valuable experience for future research.

Reference

[1] Bai, S., Song, W., Chen, J., Ji, Y., Zhong, Z., Yang, J., ... & Chen, B. (2025). Towards a unified understanding of robot manipulation: A comprehensive survey. *arXiv preprint arXiv:2510.10903*.

[2] Bayoudh, K., Knani, R., Hamdaoui, F., & Mtibaa, A. (2022). A survey on deep multimodal learning for computer vision: advances, trends, applications, and datasets. *The Visual Computer, 38*(8), 2939-2970.

[3] Black, K., Brown, N., Darpinian, J., Dhabalia, K., Driess, D., Esmail, A., ... & Zhilinsky, U. (2025). π0.5: a Vision-Language-Action Model with Open-World Generalization. In *9th Annual Conference on Robot Learning*.

[4] Bouchey, B., Castek, J., & Thygeson, J. (2021). Multimodal learning. In *Innovative learning environments in STEM higher education: Opportunities, challenges, and looking forward* (pp. 35-54). Cham: Springer International Publishing.

[5] Chen, X., Djolonga, J., Padlewski, P., Mustafa, B., Changpinyo, S., Wu, J., ... & Soricut, R. (2023). Pali-x: On scaling up a multilingual vision and language model. *arXiv preprint arXiv:2305.18565*.

[6] Din, M. U., Akram, W., Saoud, L. S., Rosell, J., & Hussain, I. (2025). Vision language action models in robotic manipulation: A systematic review. *arXiv preprint arXiv:2507.10672*.

[7] He, R., Liu, L., Ye, H., Tan, Q., Ding, B., Cheng, L., ... & Si, L. (2021). On the effectiveness of adapter-based tuning for pretrained language model adaptation. In *Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing* (volume 1: long papers) (pp. 2208-2222).

[8] Huang, D., Fang, Z., Zhang, T., Li, Y., Zhao, L., & Xia, C. (2025). Co-rft: Efficient fine-tuning of vision-language-action models through chunked offline reinforcement learning. *arXiv preprint arXiv:2508.02219*.

[9] Kawaharazuka, K., Oh, J., Yamada, J., Posner, I., & Zhu, Y. (2025). Vision-language-action models for robotics: A review towards real-world applications. *IEEE Access*.

[10] Kim, M. J., Finn, C., & Liang, P. (2025). Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*.

[11] Kim, M. J., Pertsch, K., Karamcheti, S., Xiao, T., Balakrishna, A., Nair, S., ... & Finn, C. (2024). Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*.

[12] Lu, Z. (2023). A theory of multimodal learning. *Advances in Neural Information Processing Systems, 36*, 57244-57255.

[13] Ma, Y., Song, Z., Zhuang, Y., Hao, J., & King, I. (2024). A survey on vision-language-action models for embodied ai. *arXiv preprint arXiv:2405.14093*.

[14] O'Neill, A., Rehman, A., Maddukuri, A., Gupta, A., Padalkar, A., Lee, A., ... & Chen, M. (2024). Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *2024 IEEE International Conference on Robotics and Automation* (ICRA) (pp. 6892-6903). IEEE.

[15] Sapkota, R., Cao, Y., Roumeliotis, K. I., & Karkee, M. (2025). Vision-language-action models: Concepts, progress, applications and challenges. *arXiv preprint arXiv:2505.04769*.

[16] Shao, R., Li, W., Zhang, L., Zhang, R., Liu, Z., Chen, R., & Nie, L. (2025). Large vlm-based vision-language-action models for robotic manipulation: A survey. *arXiv preprint arXiv:2508.13073*.

[17] Tang, J., Li, K., Hou, M., Jin, X., Kong, W., Ding, Y., & Zhao, Q. (2022). MMT: Multi-way Multi-modal Transformer for Multimodal Learning. In *IJCAI* (pp. 3458-3465).

[18] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

[19] Xu, P., Zhu, X., & Clifton, D. A. (2023). Multimodal learning with transformers: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 45*(10), 12113-12132.

[20] Xue, H., Huang, X., Niu, D., Liao, Q., Kragerud, T., Gravdahl, J. T., ... & Sastry, S. (2025). Leverb: Humanoid whole-body control with latent vision-language instruction. *arXiv preprint arXiv:2506.13751*.

[21] Yu, P., Xu, X., & Wang, J. (2024). Applications of large language models in multimodal learning. *Journal of Computer Technology and Applied Mathematics, 1*(4), 108-116.

[22] Yuan, Y., Li, Z., & Zhao, B. (2025). A survey of multimodal learning: Methods, applications, and future. *ACM Computing Surveys, 57*(7), 1-34.

[23] Zhang, D., Tigges, C., Zhang, Z., Biderman, S., Raginsky, M., & Ringer, T. (2024). Transformer-based models are not yet perfect at learning to emulate structural recursion. *arXiv preprint arXiv:2401.12947*.

[24] Zhang, W., Liu, H., Qi, Z., Wang, Y., Yu, X., Zhang, J., ... & Jin, X. (2025). Dreamvla: a vision-language-action model dreamed with comprehensive world knowledge. *arXiv preprint arXiv:2507.04447*.

[25] Zhang, Y., Gong, K., Zhang, K., Li, H., Qiao, Y., Ouyang, W., & Yue, X. (2023). Meta-transformer: A unified framework for multimodal learning. *arXiv preprint arXiv:2307.10802*.

[26] Zhao, Q., Lu, Y., Kim, M. J., Fu, Z., Zhang, Z., Wu, Y., ... & Xiang, D. (2025). Cot-vla: Visual chain-of-thought reasoning for vision-language-action models. In *Proceedings of the Computer Vision and Pattern Recognition Conference* (pp. 1702-1713).

[27] Zheng, Y., Zhao, Z., Li, Z., Xie, Y., Gao, M., Zhang, L., & Zhang, K. (2025). Thought Communication in Multiagent Collaboration. arXiv preprint arXiv:2510.20733.

[28] Zhong, Y., Bai, F., Cai, S., Huang, X., Chen, Z., Zhang, X., ... & Yang, Y. (2025). A Survey on Vision-Language-Action Models: An Action Tokenization Perspective. *arXiv preprint arXiv:2507.01925*.

[29] Zhou, Z., Zhu, Y., Zhu, M., Wen, J., Liu, N., Xu, Z., ... & Xu, Y. (2025). Chatvla: Unified multimodal understanding and robot control with vision-language-action model. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing* (pp. 5377-5395).

[30] Zitkovich, B., Yu, T., Xu, S., Xu, P., Xiao, T., Xia, F., ... & Han, K. (2023). Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning* (pp. 2165-2183). PMLR.