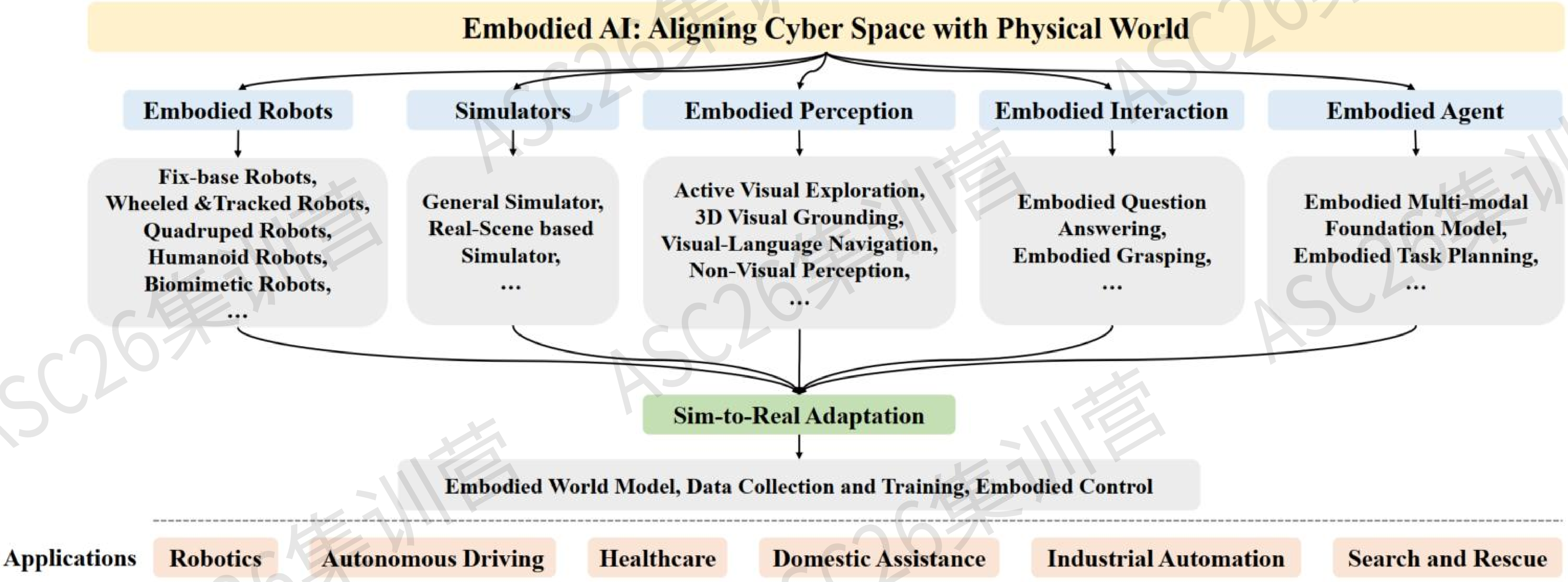
Embodied Intelligence

ASC Committee Application Expert
Zhan Gong

- **Embodied Intelligence Overview**
- Vision-Language-Action
- World Model

Embodied Intelligence Overview



Embodied Intelligence Overview

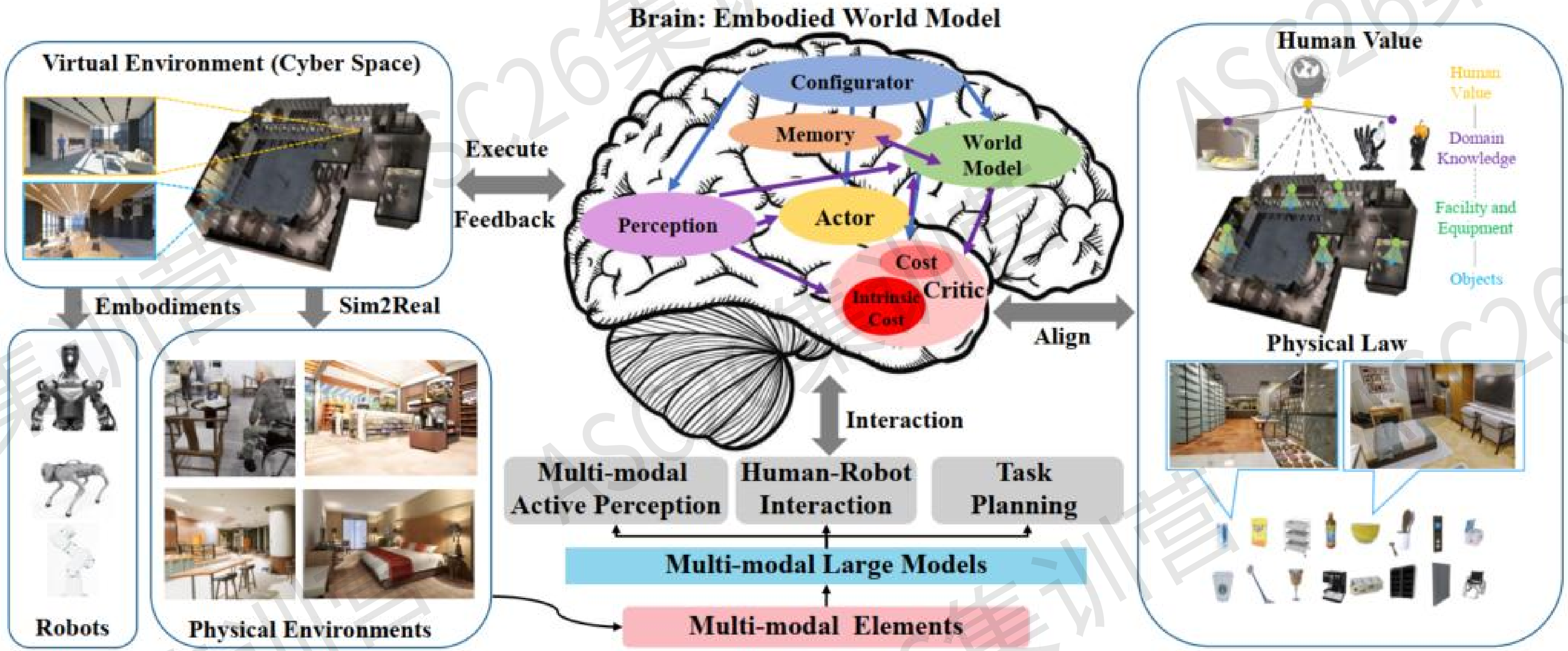


Fig. 2. The overall framework of the embodied agent based on MLMs and WMs. The embodied agent has a embodied world model as its “brain”. It has the capability to understand the virtual-physical environment and actively perceive multi-modal elements. It can fully understand human intention, align with human value, decompose complex tasks, and execute accurate actions, as well as interact with humans and utilize knowledge bases and tools.

Embodied Intelligent Robots

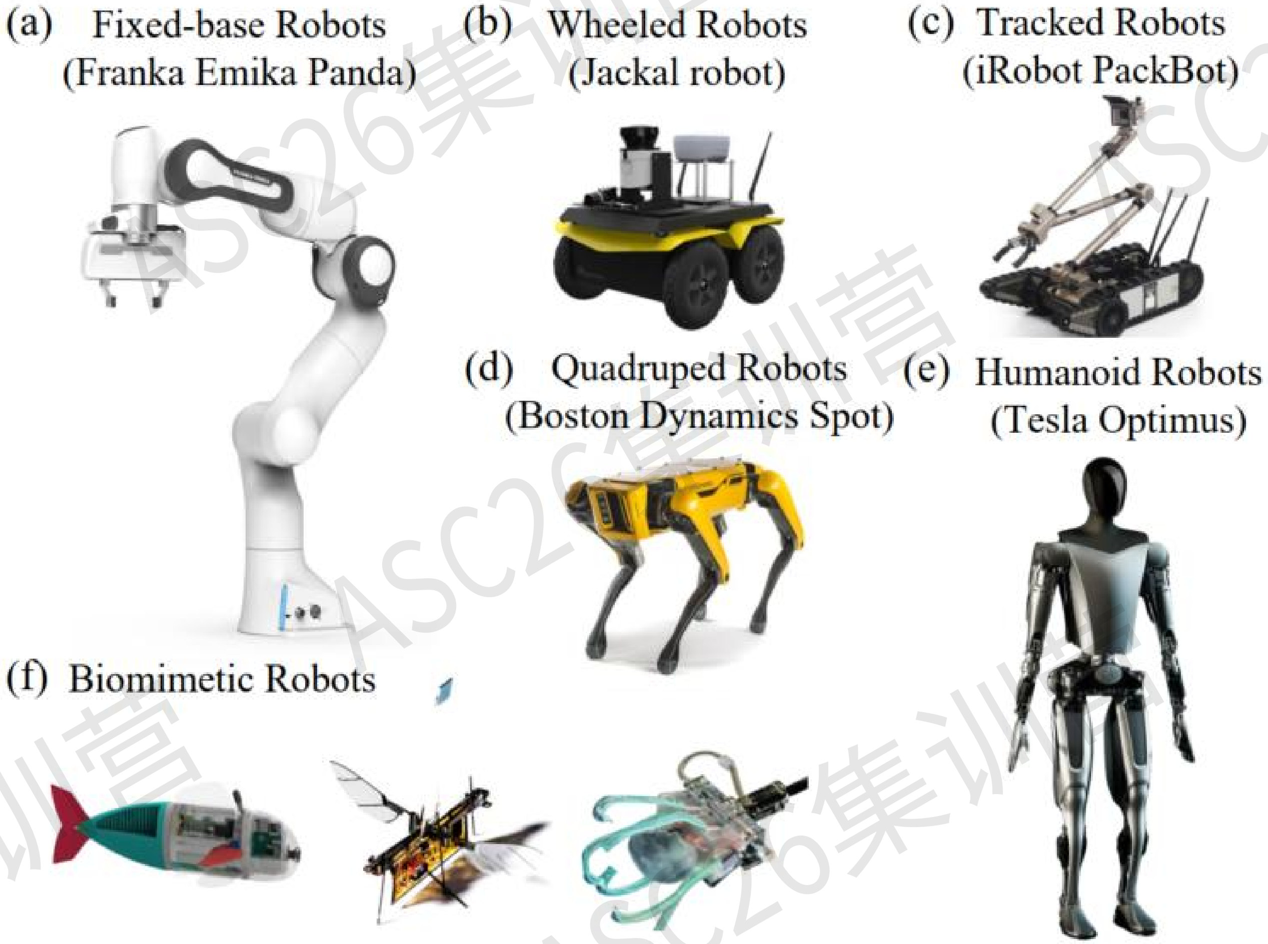


Fig. 4. The Embodied Robots include Fixed-base Robots, Quadruped Robots, Humanoid Robots, Wheeled Robots, Tracked Robots, and Biomimetic Robots.

- Embodied Intelligence Overview
- **Vision-Language-Action**
- World Model

VLA Model

The idea of VLA is to help robots interpret what they see, understand instructions, and act in the physical world. To do this, VLAs combine perception, language understanding, and control in a single system. They push robot learning toward foundation-model-style control, where just one model can handle many tasks by leveraging pretrained multimodal knowledge.

Most VLA models are built around three core components:

- **Vision-Language backbone**: VLAs typically start from a large Vision Language Model (VLM) pretrained on image–text data. VLMs already know how to recognize objects, understand text, reason spatially, and even solve math problems.
- **Action interface**: On top of the VLM, VLAs add a mechanism to produce robot actions. Depending on the design, this can be direct action prediction (continuous control), action chunks or trajectories, or structured action representations learned from demonstrations.
- **Multimodal inputs**: VLAs usually condition on camera images, natural language instructions, and often robot state like joint positions, gripper state, and others.

A good VLA model should accomplish two missions well: preserve open-world reasoning from the VLM, and correctly turn that reasoning – what a robot sees and is told – into actions.

VLA Model Evolution

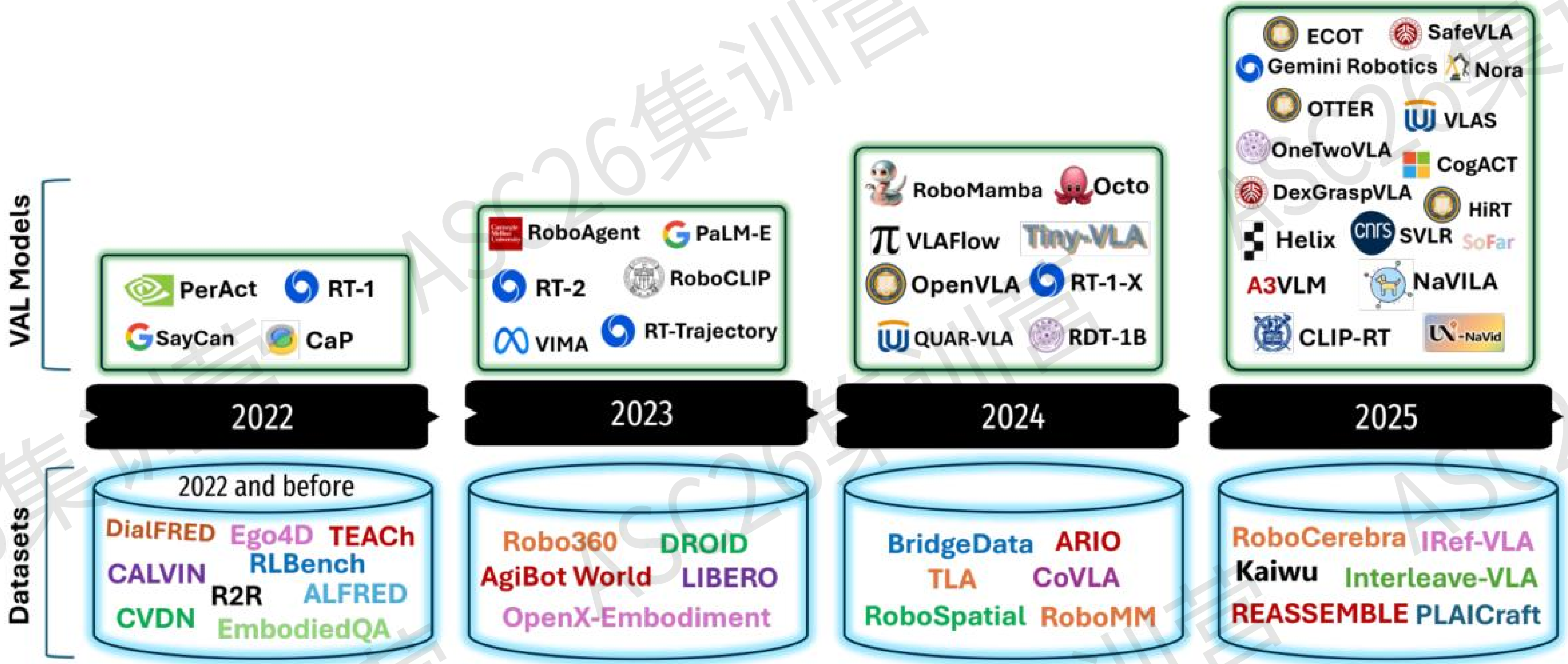


Figure 1: VLA models, datasets, and contributing institutions from 2022 to 2025. The top row presents major VLA models introduced each year, alongside their associated institutions (logos within red boxes). The bottom row displays key datasets used to train and evaluate these models, grouped by release year. The figure highlights the increasing scale and diversity of datasets and institutional involvement, with contributions from academic (e.g., CMU, CNRS, UC, Peking Uni) and industrial labs (e.g., Google, NVIDIA, Microsoft). This timeline highlights the rapid advancements in VLA research.

Classification of VLA

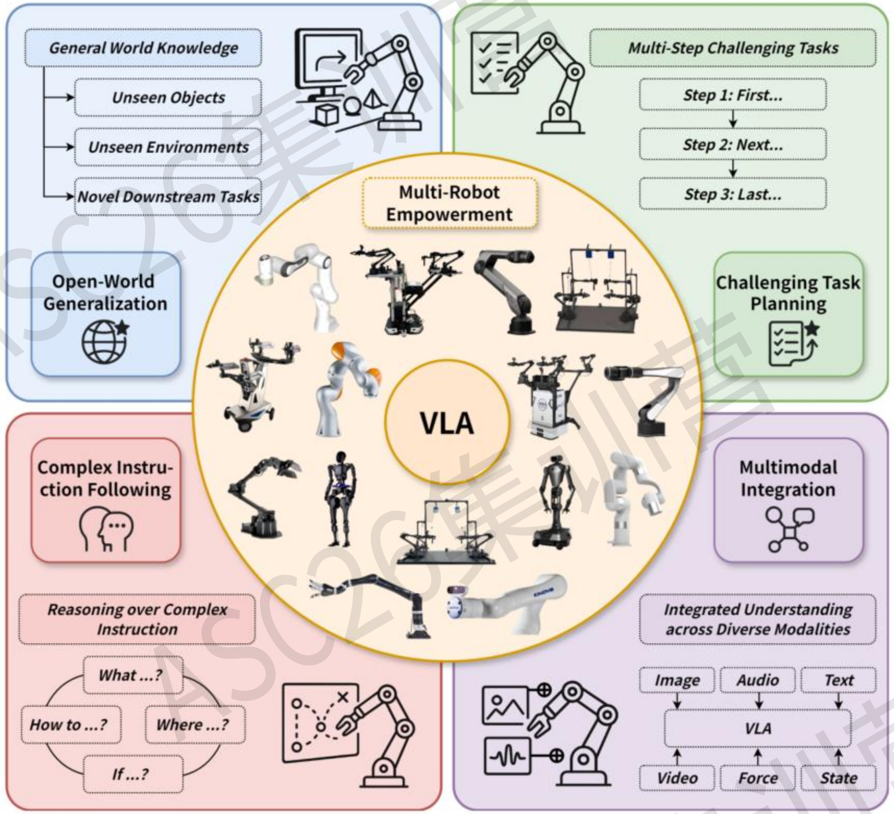


Fig. 1: Illustration of core advantages of large VLM-based Vision-Language-Action (VLA) models for robotic manipulation. Large VLM-based VLA models leverages the strengths of large Vision-Language Models (VLMs), including **(1)** open-world generalization, **(2)** hierarchical task planning, **(3)** knowledge-augmented reasoning, and **(4)** rich multimodal fusion. These capabilities empower diverse robotic arms and significantly enhance robotic intelligence.

Classification of VLA

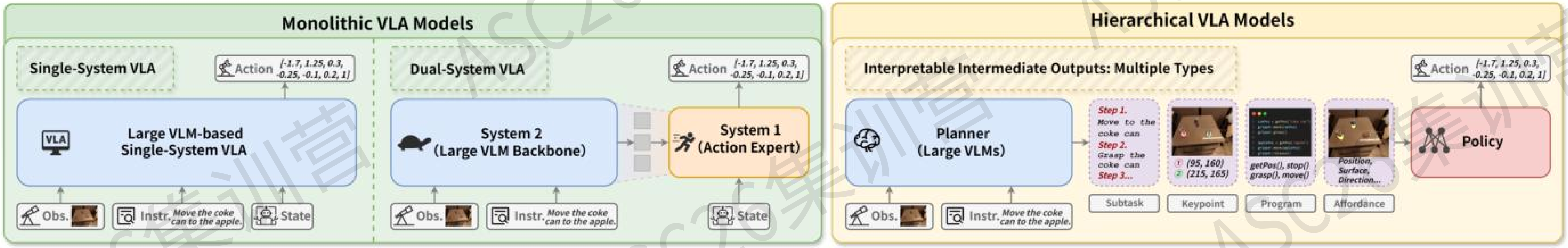


Fig. 3: Comparison of the two principal categories of large VLM-based VLA models. Monolithic models (Sec. 3) integrate perception, language understanding, and action generation within single- or dual-system architectures, with the latter incorporating an additional action expert. In contrast, hierarchical models (Sec. 4) decouple planning from policy execution through interpretable intermediate outputs (e.g., subtasks, keypoints, programs, affordances).

Classification of VLA

TABLE 1: Single-system VLA models. In the LLM / VLM column, omission of the V-Encoder indicates a VLM; otherwise, it represents an LLM. In the Learning column, "AD" denotes Autoregressive Decoding and "PD" denotes Parallel Decoding. "SFT" denotes fine-tuning distinct from action-prediction imitation learning, where tasks like captioning, VQA, reasoning and others all qualify as SFT. "A" and "B" in parentheses represent the learning methods used by Action head or Backbone.

Model	V-Encoder	LLM / VLM	Learning	Contribution
Classic Paradigm: Autoregressive Decoding				
RT-2 [27]	-	PaLI-X / PaLM-E	AD (A), SFT (B)	Represent actions as VLM tokens to enable generalization.
RT-2-X [90]	ViT-22B	UL2	AD (A), SFT (B)	Fine-tune on cross-robot data for positive skill transfer.
OpenVLA [26]	DINOv2 + SigLIP	LLaMA2-7B	AD (A)	Open-source 7B-parameter VLA model for generalist robot control.
Paradigm Derivations: Model Performance Enhancement				
LEO agent [94]	ConvNext	Vicuna-7B	AD (A), SFT (B)	Combine object-centric 3D features with LLM for action.
ECoT [95]	DINOv2 + SigLIP	LLaMA2-7B	AD (A), SFT (B)	Incorporate chain-of-thought to enhance policy explainability.
ReVLA [96]	DINOv2 + SigLIP	LLaMA2-7B	AD (A)	Reverse backbone gradually to preserve visual generalization.
TraceVLA [97]	DINOv2 + SigLIP	LLaMA2-7B	AD (A)	Propose visual trace prompting for spatiotemporal awareness.
FuSe [98]	-	PaliGemma-3B	AD (A), SFT (B)	Leverage natural language for cross-modal fine-tuning.
UniAct [99]	-	LLaVA-0.5B	PD (A)	Propose universal action space for versatile and adaptive control.
SpatialVLA [100]	SigLIP	Gemma2	AD (A)	Improve generalization via 3D encoding and action grid.
UP-VLA [101]	ViT + VQ-GAN	Phi1.5-1.3B	AD (A), SFT (B)	Propose unified training for semantic-spatial understanding.
VLAS [102]	CLIP	Vicuna-7B	AD (A), SFT (B)	Introduce voice modality to VLA and construct a paired dataset.
HybridVLA [34]	DINOv2 + SigLIP	LLaMA2-7B	Diff., AD (A)	Integrate diffusion and autoregressive policies to improve success.
CoT-VLA [33]	-	VILA-U	AD+PD(A), SFT(B)	Propose a visual chain-of-thought to improve planning.
VTLA [103]	-	Qwen2-VL-7B	AD (A)	Integrate visual and tactile inputs to improve task success.
OE-VLA [104]	SigLIP	Qwen1.5-7B	AD (A), SFT (B)	Introduce four open-ended tasks to expand interaction modalities.
ReFineVLA [105]	SigLIP	Gemma2	AD (A), SFT (B)	Propose reasoning-aware framework to fine-tune VLAs effectively.
LoHoVLA [106]	SigLIP	Gemma-2B	AD (A), SFT (B)	Address long-horizon tasks via hierarchical closed-loop control.
BridgeVLA [35]	SigLIP	Gemma	PD (A), SFT (B)	Project 3D data into 2D space for efficient action prediction
UnifiedVLA [107]	-	Emu3	AD (A), SFT (B)	Convert all input signals into tokens to build a unified model.
WorldVLA [38]	-	Chameleon	AD (A), SFT (B)	Combine world and action models for bidirectional improvement.
4D-VLA [108]	-	InternVL-4B	PD (A)	Integrate 4D spatiotemporal cues for efficient VLA pretraining.
VOTE [109]	DINOv2 + SigLIP	LLaMA2-7B	PD (A)	Introduce voting strategy to increase action prediction accuracy.
ST-VLA [110]	-	PaliGemma2	AD (A), SFT (B)	Project visual traces onto depth maps for better understanding.
Paradigm Derivations: Inference Efficiency Optimization				
RoboFlamingo [111]	ViT	MPT-1B	PD (A), SFT (B)	Decouple design to adapt open-sourced VLM for robotic control.
RoboMamba [112]	CLIP / SigLIP ViT-L	Mamba-2.8B/1.4B	PD (A), SFT (B)	Introduce the Mamba architecture to the VLA field.
DeeR-VLA [36]	CLIP ViT-L/14	MPT-1B / 7B	PD (A)	Propose dynamic early-exit to reduce inference overhead.
OpenVLA-OFT [44]	DINOv2 + SigLIP	LLaMA2-7B	PD (A)	Boost performance via OpenVLA-based fine-tuning.
PD-VLA [113]	CLIP ViT-L	Vicuna1.5-7B	PD (A)	Introduce parallel decoding manner for faster robot control.
MoLe-VLA [114]	DINOv2 + SigLIP	LLaMA2-7B	PD (A)	Reduce computation via dynamic LLM layer activation.
NORA [45]	-	Qwen2.5-VL-3B	AD (A)	Build efficient low-parameter model to boost performance.
FLashVLA [115]	DINOv2 + SigLIP	LLaMA	AD (A)	Propose retraining-free acceleration to improve VLA inference.
BitVLA [116]	SigLIP b1.58	BitNet b1.58 2B4T	PD (A), SFT (B)	Build ternary weight model to reduce deployment memory cost.
Spec-VLA [117]	DINOv2 + SigLIP	LLaMA2-7B	PD (A)	Propose speculative decoding to speed up without success drop.

TABLE 2: Dual-system VLA models. The "System 2 Backbone" column lists the VLM backbone used as the System 2 component in dual-system methods. The "System 1 Learning" column lists the learning methods used by the action experts as System 1. "Diff." denotes diffusion-based learning, "FM" denotes flow-matching, "MSE" denotes mean squared error, "BCE" denotes binary cross-entropy, and "AR" denotes autoregressive learning.

Model	System 2 Backbone	System 1 Learning	Contribution
Cascade-based			
DP-VLA [124]	OpenVLA	Regression	Propose a dual-system architecture for robot manipulation with efficiency and performance.
RoboDual [125]	OpenVLA	Diff.	Combine a VLA-based generalist for reasoning and a DIT specialist for control.
LCB [126]	LLaVA	Diff.	Leverage an added special token to encode VLM reasoning and act as conditions for policy.
GR00T N1 [32]	Eagle-2	FM	Combine a VLM and DiT for humanoid robots manipulation.
CogACT [127]	OpenVLA	Diff.	Propose an action ensemble algorithm to integrate the action diffusion process into VLA.
HiRT [128]	InstructBLIP	Regression	Propose a dual-system model with System 2 running at a lower frequency.
Fast-in-Slow [40]	Prismatic	Diff., AR	Propose a unified dual-system model that embeds fast execution within a VLM-based reasoner.
OpenHelix [58]	LLaVA	Diff.	Conduct auxiliary training on the token bridging VLM and policy.
ChatVLA [129]	Qwen2-VL	Diff.	Unifie vision-language-action via MoE-shared attention with separate perception/control FFNs.
ChatVLA-2 [130]	Qwen2-VL	Diff.	Enable open-world robotic reasoning via dynamic MoE routing and Reasoning-Following MLP.
Diffusion-VLA [131]	Qwen2-VL	Diff.	Merge Qwen2-VL reasoning with diffusion actions via FiLM-modulated reasoning injection.
TriVLA [132]	Eagle-2	Diff.	Introduce a world-dynamics perception module as system 3 to complement static perception.
GF-VLA [133]	LLaMA 2	Regression	Enable interpretable bimanual manipulation via information-theoretic graphs from human videos.
RationalVLA [134]	LLaVA-v1.5	Diff.	Introduce a learnable latent interface to enable instruction rejection for robust manipulation.
VQ-VLA [135]	OpenVLA	VQ-VAE	Develop a vector quantization-based action tokenizer for efficient and smoother control.
TinyVLA [136]	LLaVA	Diff.	Demonstrate that high-performance VLAs require no large-scale robotic pretraining.
Parallel-based			
π_0 [29]	PaliGemma	FM	Combine a pre-trained Vision-Language Model with a Flow Matching-based Action Expert.
π_0-FAST [123]	π_0	AR	Propose a DCT-based action tokenization enabling efficient autoregressive VLA training.
$\pi_{0.5}$ [30]	PaliGemma	FM	Convert high-level prompts into more fine-grained subtask predictions before feeding into π_0
$\pi_{0.5}$-KI [37]	PaliGemma	FM	Prevent gradients from the action expert from flowing into the VLM backbone during training.
ForceVLA [137]	π_0	Diff.	Treat force sensing as a first-class modality via MoE, improving contact-rich manipulation.
SmolVLA [31]	SmolVLM-2	FM	Propose a lightweight VLA with frozen SmolVLM-2 and flow-matching transformer.
OneTwoVLA [138]	π_0	FM	Integrate acting/reasoning in shared VLA backbone processing multi-view inputs.
Tactile-VLA [139]	π_0	FM	Integrate tactile sensing to enable force-aware, generalizable contact-rich manipulation.
GR-3 [140]	Qwen2.5-VL	FM	Combine VL data and few-shot trajectories for robust manipulation in long-horizon or unseen tasks.
villa-X [141]	PaliGemma	FM	Integrate proprioceptively grounded latent actions and robot actions in a joint diffusion process.

Classification of VLA

TABLE 3: Hierarchical VLA models. The "Type" column denotes the output type of the planner, where "K" represents Keypoint, "S" represents Subtask, and "P" represents Program. The "Learning" column specifies the learning method adopted by the model, where "SFT" refers to Supervised Fine-Tuning, "RL" denotes Reinforcement Learning, "IM" indicates Imitation Learning, and "API" is a special case referring to the invocation of pre-existing models.

Model	Type	Backbone	Learning	Contribution
Planner-Only				
MoManipVLA [146]	K	OpenVLA-7B	IM	Leverage VLA models to predict waypoints and optimize full-body trajectories.
ManipLVM-R1 [46]	K	Qwen2.5-VL-3B	RL	GRPO tuning for affordance and trajectory, robust performance in OOD situations.
PaLM-E [85]	S	PaLM	SFT	Train a VLM capable of general VQA and robot manipulation instruction generation.
Emb-Reas [47]	S	Qwen2-VL-7B	SFT	Construct a VLM and open-sourced dataset with planning, reasoning, and reflection.
RoboPoint [147]	K	Vicuna-v1.5-13B	SFT	Finetune VLM for spatial affordance prediction in the form of keypoints.
Reinforced [148]	S	Qwen2.5-VL-7B	SFT, RL	Conduct GRPO on a finetuned model, bringing better generalization to unseen.
CoM [149]	P	Gemini 1.5 Pro	API	Sequential multimodal prompting to extract force-aware manipulation from demos.
RoVI [150]	K, P	GPT-4o / LLaVA-13B	SFT	Visual sketch-based instruction and hierarchical pipeline for precise manipulation.
ReLEP [151]	P	LLaVA-1.6-7B	SFT	A planning framework with implicit logical inference and hallucination mitigation.
ViLa [152]	S	GPT-4V	API	A VLM planner integrating perception and reasoning without affordance models.
RoboBrain [153]	K, S	LLaVA	SFT	Provide a hierarchical VLA focus on planning, affordance, and trajectory.
Planner+Policy				
HAMSTER [48]	K	VILA-1.5-13B	SFT, IM	Propose an out-of-the-box way for trajectory prediction to assist the low-level policy.
HiRobot [154]	S	PaliGemma-3B, π_0	SFT, IM	A hierarchical VLA with high explainability and capacity for complex tasks.
Agentic Robot [155]	S	GPT-4o	SFT	A closed-loop hierarchical pipeline where a VLM is attached to a completion judge.
DexVLA [156]	S	Qwen2-VL	SFT, IM	Combine a VLM with a large diffusion head up to 1B and a 3-stage training recipe.
Instruct2Act [157]	P	ChatGPT	API	Generate programs that call APIs for mapping from instructions to actions.
RoboMatrix [158]	S	Vicuna 1.5	SFT	VLA with modular scheduling layer, skill layer, and hardware layer.
PointVLA [159]	S	Qwen2-VL	SFT	Attach VLA with a point cloud encoder and injector to equip spatial perception.
A_0 [49]	K	Qwen2.5-7B	SFT, API	Hierarchical affordance-aware diffusion with embody-agnostic keypoint prediction.
FSD [160]	S	CLIP, Vicuna	SFT	Propose the generation of visual aids via SrCoT for zero-shot manipulation.
RoBridge [161]	S	GPT-4o	IM, RL	Bridge VLM cognition with RL execution via invariant operable representation.
Robocerebra [162]	S	GPT-4o, Qwen2.5-VL	SFT	A novel benchmark and hierarchical framework for long-horizontal evaluation.
DexGraspVLA [163]	S	Qwen-VL	API, IM	Combine a VLM as a high-level planner with a low-level diffusion-based policy.
RT-H [28]	S	PaLI-X 55B	SFT, IM	An action hierarchy architecture using language motion as a middle representation.
ReKep [50]	K, P	GPT-4o	API	Training-free trajectory generation by keypoint constraints for manipulation.
VoxPoser [164]	P	GPT-4	API	Propose language-guided 3D value maps with zero-shot generalization capabilities.
SkillDiffuser [165]	S	Transformer	IM	A hierarchical VLA with high-level model and low-level model.
RT-Affordance [166]	K	PaLM-E 2	SFT, IM	Use visual affordances as intermediate features for web-robot knowledge transfer.
HiBerNAC [167]	S	PaLM2	SFT	Propose an asynchronous model that mimics the hierarchical structure of the brain
LLARVA [168]	K	Llama 2 7B	SFT, IM	A vision-action instruction tuning paradigm and a large instruction tuning dataset.
MALMM [169]	S, P	GPT-4-Turbo	API	Three-agent system combining planner, supervisor, and coder without post-training.
VLA-Touch [170]	S	GPT-4o	IM	Integrate tactile sensing into VLA control via diffusion-based imitation learning.

OpenVLA（Single-system） Stanford, 2024.09

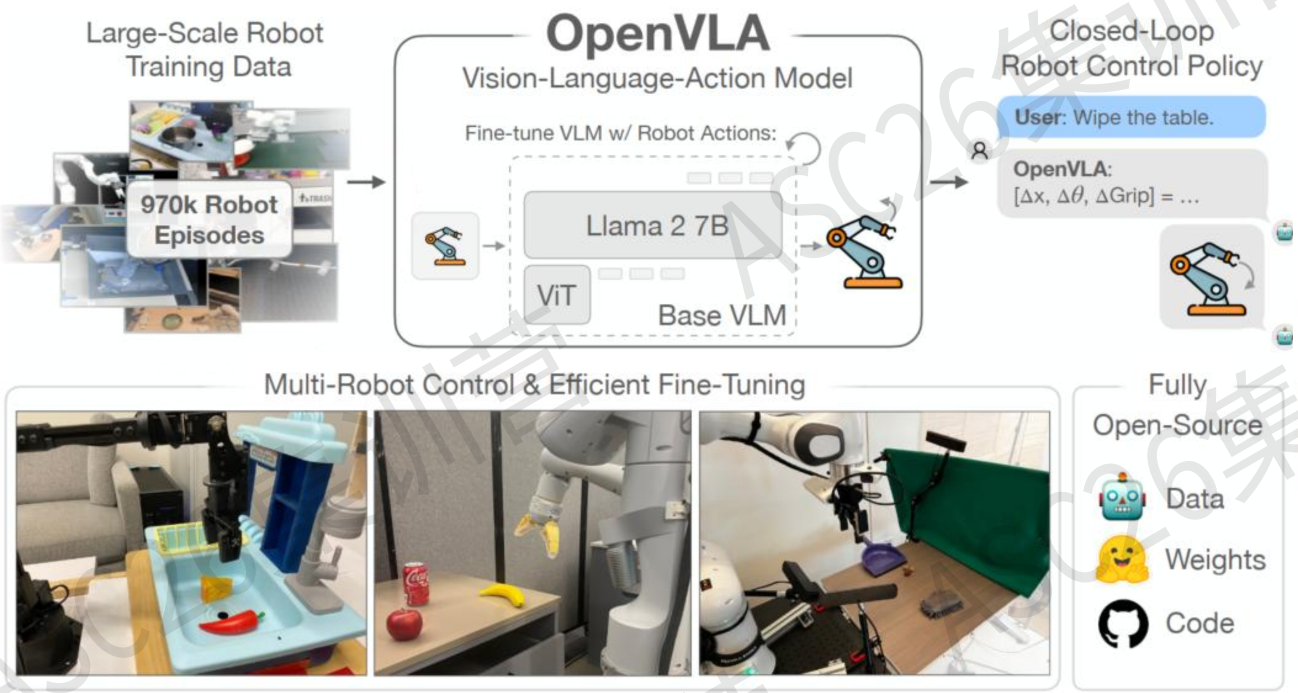


Figure 1: We present OpenVLA, a 7B-parameter open-source vision-language-action model (VLA), trained on 970k robot episodes from the Open X-Embodiment dataset [1]. OpenVLA sets a new state of the art for generalist robot manipulation policies. It supports controlling multiple robots out of the box and can be quickly adapted to new robot domains via parameter-efficient fine-tuning. The OpenVLA checkpoints and PyTorch training pipeline are fully open-source and models can be downloaded and fine-tuned from HuggingFace.

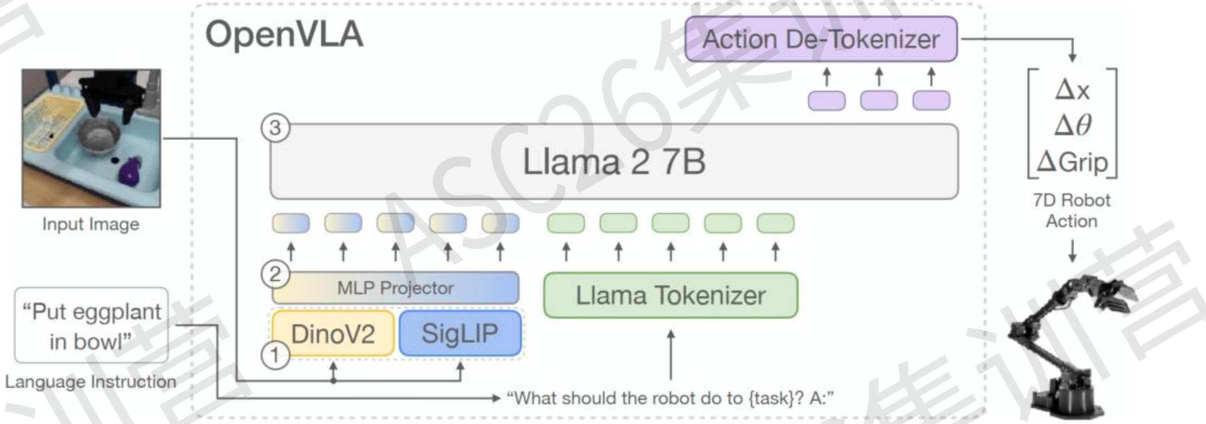


Figure 2: **OpenVLA model architecture.** Given an image observation and a language instruction, the model predicts 7-dimensional robot control actions. The architecture consists of three key components: (1) a **vision encoder** that concatenates Dino V2 [25] and SigLIP [79] features, (2) a **projector** that maps visual features to the language embedding space, and (3) the **LLM backbone**, a Llama 2 7B-parameter large language model [10].

3.5 Infrastructure for Training and Inference

The final OpenVLA model is trained on a cluster of 64 A100 GPUs for 14 days, or a total of 21,500 A100-hours, using a batch size of 2048. During inference, OpenVLA requires 15GB of GPU memory when loaded in `bfloat16` precision (i.e., without quantization) and runs at approximately 6Hz on one NVIDIA RTX 4090 GPU (without compilation, speculative decoding, or other inference speed-up tricks). We can further reduce the memory footprint of OpenVLA during inference via quantization, without compromising performance in real-world robotics tasks, as shown in Section 5.4. We report inference speed on various consumer- and server-grade GPUs in Fig. 6. For convenience, we implement a remote VLA inference server to allow real-time remote streaming of action predictions to the robot – removing the requirement of having access to a powerful local compute device to control the robot. We release this remote inference solution as part of our open-source code release (Section 4).

GR00T N1（Dual-system） NVIDIA, 2025.03

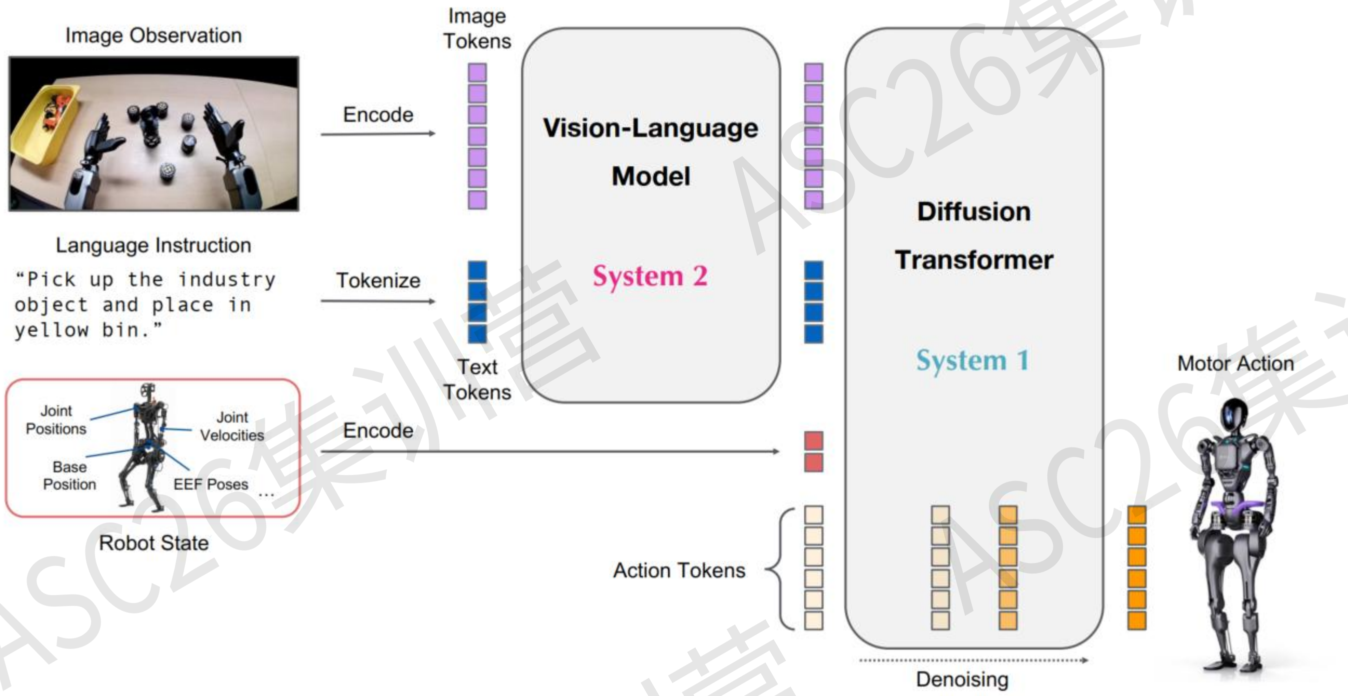


Figure 2: **GR00T N1 Model Overview.** Our model is a Vision-Language-Action (VLA) model that adopts a dual-system design. We convert the image observation and language instruction into a sequence of tokens to be processed by the Vision-Language Model (VLM) backbone. The VLM outputs, together with robot state and action encodings, are passed to the Diffusion Transformer module to generate motor actions.

Training Infrastructure

We train GR00T N1 on a cluster managed via NVIDIA OSMO (NVIDIA, 2025), an orchestration platform for scaling complex robotics workloads. The training cluster is equipped with H100 NVIDIA GPUs connected via NVIDIA Quantum-2 InfiniBand in a fat-tree topology. We facilitate fault-tolerant multi-node training and data ingestion via a custom library built on top of the Ray distributed computing library (Moritz et al., 2018). We use up to 1024 GPUs for a single model. GR00T-N1-2B used roughly 50,000 H100 GPU hours for pretraining.

Compute-constrained finetuning was tested in the context of a single A6000 GPU. If only tuning the adapter layers (action and state encoders + action decoder) and DiT, a batch size up to 200 can be used. When tuning the vision encoder, a batch size of up to 16 can be used.

2. GR00T N1 Foundation Model

GR00T N1 is a Vision-Language-Action (VLA) model for humanoid robots trained on diverse data sources. The model contains a vision-language backbone that encodes language and image input and a DiT-based flow-matching policy that outputs high-frequency actions. We use the NVIDIA Eagle-2 VLM (Li et al., 2025) as the vision-language backbone. Specifically, our publicly released GR00T-N1-2B model has 2.2B parameters in total, with 1.34B in the VLM. The inference time for sampling a chunk of 16 actions is 63.9ms on an L40 GPU using bf16. Fig. 2 provides a high-level overview of our model design. We highlight three key features of GR00T N1:

Challenges

- Architecture
  - **Memory and Long-Term Planning**: Existing models lack explicit memory mechanisms, making it difficult to handle planning and execution of long-sequence and multi-step tasks, as well as processing historical context.
  - **3D and 4D Perception**: There is a need to extract precise 3D and 4D (spatiotemporal) information from 2D image inputs to support accurate manipulation.
  - **Model Efficiency**: VLMs have high computational costs and slow inference speeds, which are insufficient for real-time robot control requirements.
- Data
  - **Reality Gap**: Simulation datasets lack the visual complexity of real environments, while collecting real-world data is expensive and limited in scale.
  - **Modality Imbalance**: Most datasets primarily provide RGB images and text, lacking critical sensor modalities such as depth maps, force/torque, and tactile data.
  - **Data Fragmentation**: There is a lack of unified, large-scale, cross-scenario embodied AI datasets, particularly gaps in high task complexity and multimodal richness.
- Benchmark
  - Existing benchmarks mostly focus on short-horizon pick-and-place tasks and report simple success rates, which are insufficient to evaluate practical challenges like long-term planning.

- Embodied Intelligence Overview
- Vision-Language-Action
- **World Model**

World Model

A world model in the context of embodied intelligence is a learnable model that simulates changes in environmental states internally to predict future outcomes. It serves as a bridge connecting embodiment and intelligence, constructed through the agent's sensory-motor interactions with the environment (e.g., touch, vision) and enabling adaptive behavior in complex scenarios.

For embodied AI systems, world models must not only generate static scene descriptors but also support actionable predictions, ensuring physically compliant interactions by modeling the dynamics of the external world. This integration of perception, cognition, and predictive simulation distinguishes it from purely generative visual models, making it a core component for AGI by unifying semantic reasoning (via large language models) and physical interaction constraints.

World Model

TABLE 1
A summary of representative world models in robotics and general-purpose domains.

Paper	Publication	Taxonomy[1]	Characteristics[2]	Datasets Platform: DMC	Atari	RLBench	SSv2	OXE	Meta-world	Franka	RT-1	LIBERO	Other(s)	Total[3]	Modality: RGB	Action	Proprio.	Depth	Language	Other(s)	Reality[4]
PlaNet [38]	ICML'19	Dec/Seq/GLV	RSSM	✓										1	✓	✓					
Dreamer [10]	ICLR'20	Dec/Seq/GLV	RSSM	✓	✓								✓	3	✓	✓					
GLAMOR [39]	ICLR'21	Dec/Seq/GLV	IDM	✓	✓									2	✓	✓					
DreamerV2 [11]	ICLR'21	Dec/Seq/GLV	RSSM	✓	✓									2	✓	✓					
TransDreamer [28]	arXiv'22	Dec/Seq/GLV	TSSM	✓	✓								✓	4	✓	✓					
Iso-Dream [40]	NeurIPS'22	Dec/Seq/GLV	IDM	✓									✓	4	✓	✓					
MWM [41]	CoRL'22	Dec/Seq/TFS	RSSM	✓		✓			✓					3	✓	✓					
Inner Monologue [42]	CoRL'22	Dec/Seq/TFS	CoT										✓	3	✓	✓			✓		✓
DayDreamer [43]	CoRL'22	Dec/Seq/GLV	RSSM										✓	4	✓	✓	✓	✓			✓
TWM [29]	ICLR'23	Dec/Seq/TFS	Transformer		✓									1	✓	✓					
IRIS [44]	ICLR'23	Dec/Seq/TFS	Transformer		✓									1	✓	✓					
WorldDreamer [45]	arXiv'24	Gen/Glo/TFS	Transformer										✓	4	✓	✓			✓		
Statler [46]	ICRA'24	Dec/Seq/TFS	LLM										✓	2	✓	✓		✓	✓		✓
Pandora [47]	arXiv'24	Gen/Seq/TFS	Video Diffusion				✓						✓	2	✓				✓		
DWL [48]	RSS'24	Dec/Seq/GLV	MLP										✓	4		✓	✓			✓	✓
RoboDreamer [49]	ICML'24	Dec/Glo/TFS	IDM			✓					✓			2	✓	✓			✓	✓	
Genie [50]	ICML'24	Gen/Seq/TFS	Transformer								✓		✓	3	✓	✓					
V-JEPA [51]	TMLR'24	Gen/Glo/TFS	JEPA				✓						✓	6	✓						
PreLAR [52]	ECCV'24	Dec/Seq/GLV	RSSM			✓	✓		✓					3	✓	✓					
ManiGaussian [53]	ECCV'24	Dec/Seq/DRR	3DGS			✓								1	✓	✓	✓	✓	✓		
ECoT [54]	CoRL'24	Dec/Glo/TFS	CoT										✓	3	✓	✓			✓		✓
VidMan [55]	NeurIPS'24	Dec/Glo/TFS	IDM			✓		✓					✓	4	✓	✓	✓		✓		
iVideoGPT [56]	NeurIPS'24	Gen/Seq/TFS	Transformer				✓	✓	✓				✓	6	✓	✓					
EnerVerse [34]	arXiv'25	Dec/Seq/SLG	Video Diffusion								✓	✓	✓	4	✓	✓			✓		✓
GLAM [57]	AAAI'25	Dec/Seq/GLV	Mamba		✓									1	✓	✓					
NavCoT [58]	TPAMI'25	Dec/Seq/TFS	CoT										✓	4	✓	✓			✓		
DreamerV3 [12]	Nature'25	Dec/Seq/GLV	RSSM	✓	✓								✓	8	✓	✓	✓				
MineWorld [59]	arXiv'25	Dec/Seq/TFS	Transformer										✓	1	✓	✓					
DreMa [60]	ICLR'25	Dec/Seq/DRR	3DGS			✓				✓				2	✓	✓		✓	✓	✓	✓
S2-SSM [61]	arXiv'25	Gen/Seq/TFS	Mamba										✓	1	✓					✓	
RLVR-World [62]	arXiv'25	Gen/Seq/TFS	RLVR							✓			✓	3	✓	✓			✓		
StateSpaceDiffuser [63]	arXiv'25	Gen/Seq/TFS	Mamba										✓	2	✓	✓					
DeepVerse [64]	arXiv'25	Gen/Seq/TFS	DiT										✓	1	✓			✓	✓	✓	
ORV [65]	arXiv'25	Gen/Glo/SLG	DiT								✓		✓	4	✓	✓		✓	✓	✓	
V-JEPA 2 [14]	arXiv'25	Gen/Glo/TFS	JEPA				✓			✓			✓	15	✓	✓	✓		✓		✓
NWM [66]	CVPR'25	Dec/Seq/TFS	DiT										✓	6	✓	✓					
WorldVLA [67]	arXiv'25	Dec/Seq/TFS	Transformer									✓		1	✓	✓			✓		
World4Omni [68]	arXiv'25	Gen/Seq/TFS	VLM			✓							✓	2	✓	✓			✓	✓	✓
Dyn-O [69]	arXiv'25	Dec/Seq/TFS	Mamba										✓	1	✓	✓					
DINO-WM [70]	ICML'25	Dec/Seq/SLG	Transformer	✓									✓	3	✓	✓	✓				
EVA [71]	ICML'25	Gen/Seq/TFS	RoG					✓			✓		✓	4	✓	✓			✓		
AdaWorld [72]	ICML'25	Gen/Seq/TFS	Video Diffusion				✓	✓				✓	✓	6	✓	✓					
MindJourney [73]	arXiv'25	Gen/Seq/SLG	VLM										✓	2	✓	✓			✓		
GAF [74]	arXiv'25	Dec/Seq/DRR	4DGS			✓								1	✓	✓	✓			✓	
Yume [75]	arXiv'25	Gen/Seq/TFS	DiT										✓	1	✓	✓			✓		
villa-X [76]	arXiv'25	Dec/Glo/TFS	IDM				✓	✓			✓	✓	✓	5	✓	✓	✓		✓		✓
AETHER [77]	ICCV'25	Gen/Glo/SLG	DiT										✓	6	✓	✓		✓		✓	
TesserAct [78]	ICCV'25	Dec/Glo/SLG	IDM			✓	✓				✓		✓	4	✓			✓	✓	✓	
MineDreamer [79]	IROS'25	Dec/Seq/TFS	CoI										✓	3	✓	✓			✓		
ManiGaussian++ [80]	IROS'25	Dec/Seq/DRR	3DGS			✓							✓	2	✓	✓	✓	✓	✓		✓

[1] **Taxonomy**: Abbreviations for the taxonomy categories defined in §3.
[2] **Characteristics**: Representative backbone or core technical approach.
[3] **Total**: Number of data platforms used. Underlined entries denote newly proposed or aggregated datasets.
[4] **Reality**: The check mark (✓) indicates validation on a physical robot.

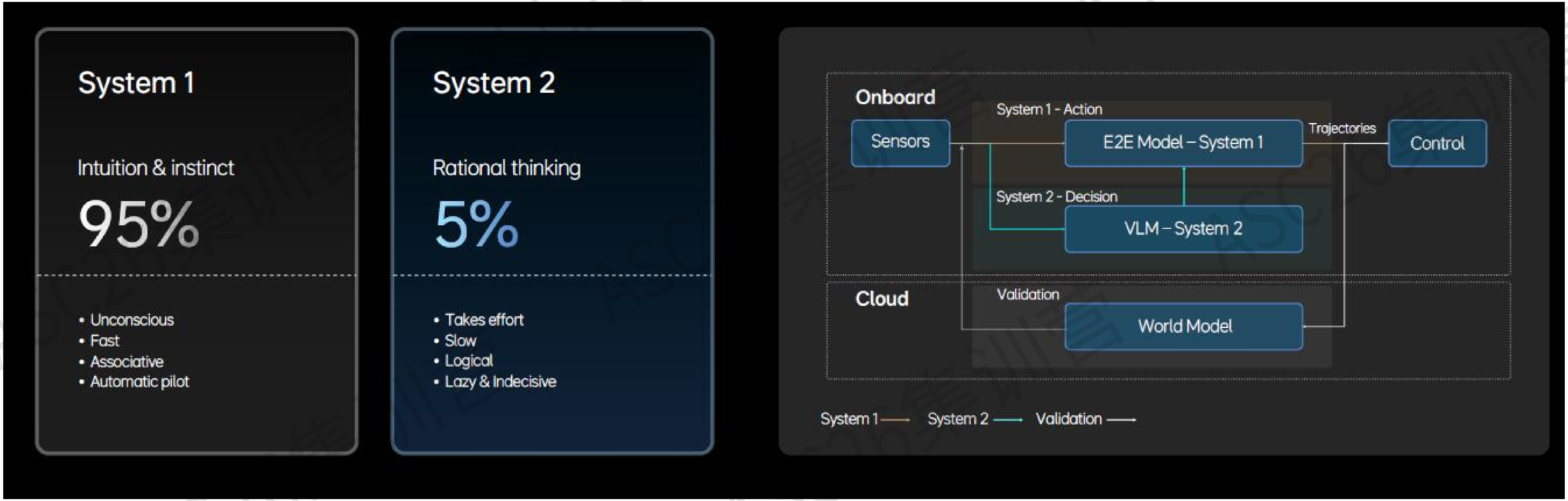
TABLE 2
A summary of representative world models for the autonomous driving domain.

Paper	Publication	Taxonomy[1]	Characteristics[2]	Datasets Platform: CARLA	nuScenes	nuPlan	Waymo	Occ3D	OpenDV	Other(s)	Total[3]	Input Modality: RGB	Motion	Map	LiDAR	Bound box	Language	Occpancy	Other(s)
MILE [81]	NeurIPS'22	Dec/Seq/GLV	RSSM	✓							1	✓	✓	✓					
Copilot4D [82]	ICLR'24	Gen/Seq/SLG	Video Diffusion		✓					✓	3		✓		✓				
SEM2 [83]	TITS'24	Dec/Seq/GLV	RSSM	✓							1	✓	✓	✓	✓				
MagicDrive3D [84]	arXiv'24	Gen/Glo/DRR	3DGS		✓						1	✓	✓	✓		✓	✓		✓
OccSora [85]	arXiv'24	Gen/Glo/SLG	Diffusion		✓			✓			2		✓					✓	
Delphi [86]	arXiv'24	Gen/Seq/SLG	Video Diffusion		✓						1	✓		✓		✓	✓		
DriveWorld [87]	CVPR'24	Dec/Seq/SLG	RSSM		✓					✓	2	✓	✓				✓	✓	
Drive-WM [88]	CVPR'24	Dec/Glo/SLG	Video Diffusion		✓						1	✓	✓	✓		✓	✓		✓
ViDAR [89]	CVPR'24	Gen/Seq/SLG	Transformer		✓						1	✓	✓		✓				
GenAD [90]	CVPR'24	Gen/Seq/TFS	Video Diffusion		✓	✓	✓		✓	✓	4	✓	✓				✓		
OccLLaMA [18]	arXiv'24	Dec/Seq/SLG	Transformer		✓			✓		✓	3	✓	✓				✓	✓	
DriveDreamer [91]	ECCV'24	Dec/Seq/SLG	GRU		✓						1	✓	✓	✓		✓	✓		
GenAD [92]	ECCV'24	Dec/Seq/SLG	GRU		✓						1	✓							
OccWorld [93]	ECCV'24	Dec/Seq/SLG	Transformer		✓			✓			2	✓	✓		✓			✓	
DOME [94]	arXiv'24	Gen/Seq/SLG	DiT		✓			✓			2	✓	✓					✓	
TOKEN [95]	CoRL'24	Dec/Glo/TFS	Transformer		✓					✓	2	✓	✓	✓			✓		
Vista [96]	NeurIPS'24	Gen/Seq/SLG	Video Diffusion		✓		✓		✓	✓	4	✓	✓				✓		
DriveDreamer-2 [97]	AAAI'25	Gen/Glo/SLG	Video Diffusion		✓						1	✓	✓	✓		✓	✓		
DTT [98]	arXiv'25	Dec/Seq/DRR	Transformer		✓			✓			2	✓	✓				✓	✓	
DynamicCity [99]	ICLR'25	Gen/Glo/SLG	DiT	✓	✓		✓	✓			4	✓	✓				✓	✓	
LidarDM [100]	ICRA'25	Gen/Seq/SLG	Diffusion				✓			✓	3		✓	✓					
FutureSightDrive [101]	arXiv'25	Dec/Seq/TFS	CoT(VLM)		✓					✓	3	✓	✓				✓		
GEM [102]	CVPR'25	Gen/Seq/SLG	Video Diffusion		✓				✓		1	✓	✓						✓
GaussianWorld [103]	CVPR'25	Gen/Seq/DRR	Transformer		✓					✓	2	✓	✓						
MaskGWM [104]	CVPR'25	Gen/Glo/TFS	DiT		✓		✓		✓		3	✓	✓				✓		
DriveDreamer4D [105]	CVPR'25	Gen/Glo/DRR	4DGS		✓	✓			✓	3	✓		✓	✓	✓	✓	✓		✓
ReconDreamer [106]	CVPR'25	Gen/Glo/DRR	3DGS		✓		✓			✓	3	✓	✓	✓	✓	✓			
WoTE [107]	ICCV'25	Dec/Seq/SLG	Transformer	✓		✓					2	✓	✓		✓				
HERMES [108]	ICCV'25	Gen/Glo/SLG	LLM		✓					✓	4	✓	✓				✓		
InfiniCube [22]	ICCV'25	Gen/Seq/DRR	3DGS				✓				1	✓	✓	✓		✓	✓		✓
DriVerse [109]	ACMMM'25	Gen/Seq/TFS	DiT		✓		✓				2	✓	✓				✓		✓

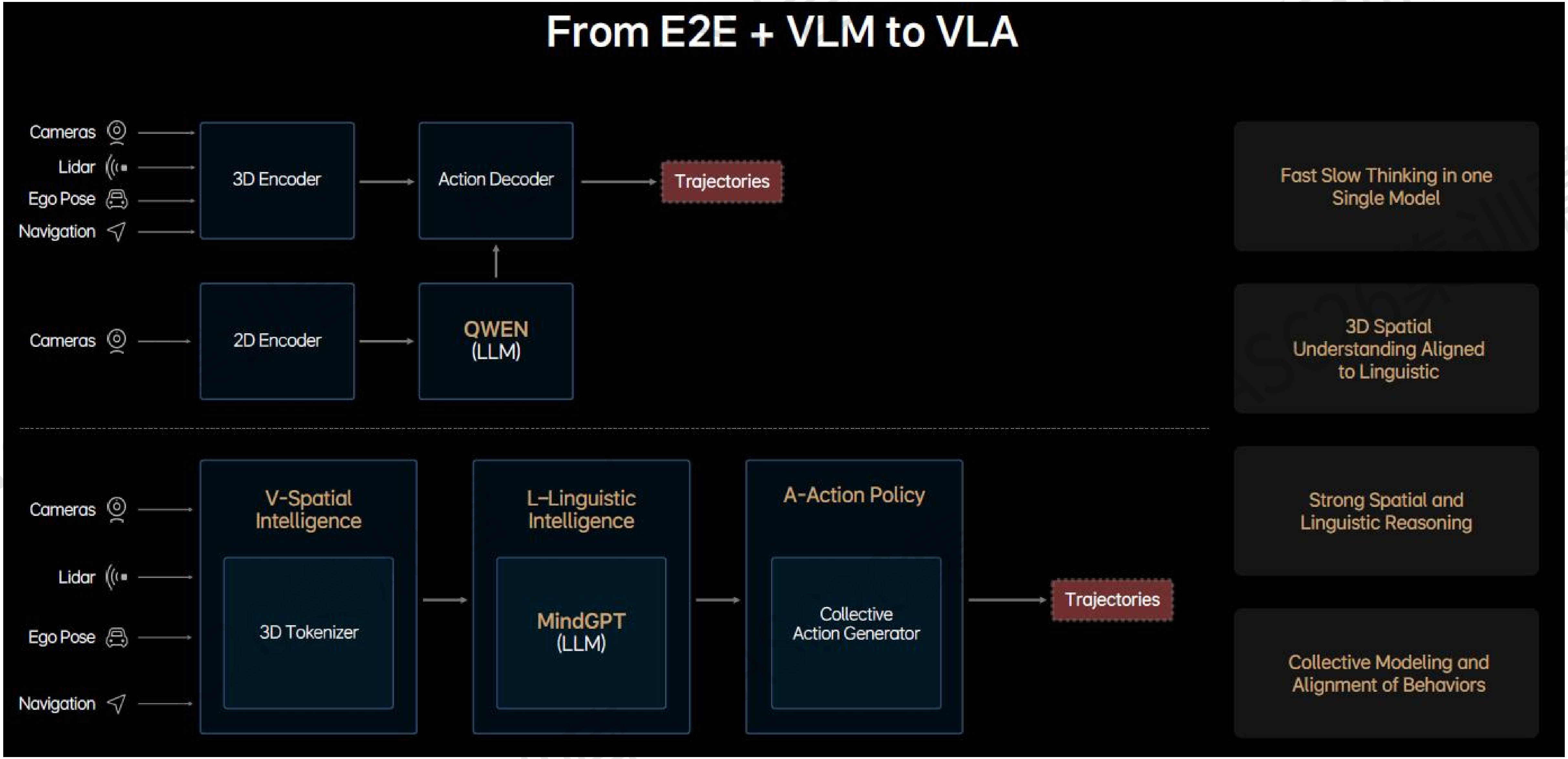
[1] **Taxonomy**: Abbreviations for the taxonomy categories defined in §3.
[2] **Characteristics**: Representative backbone or core technical approach.
[3] **Total**: Number of data platforms used. Underlined entries denote newly proposed or aggregated datasets.

Yue Ma, et al. Controllable Video Generation: A Survey. arXiv:2507.16869v1

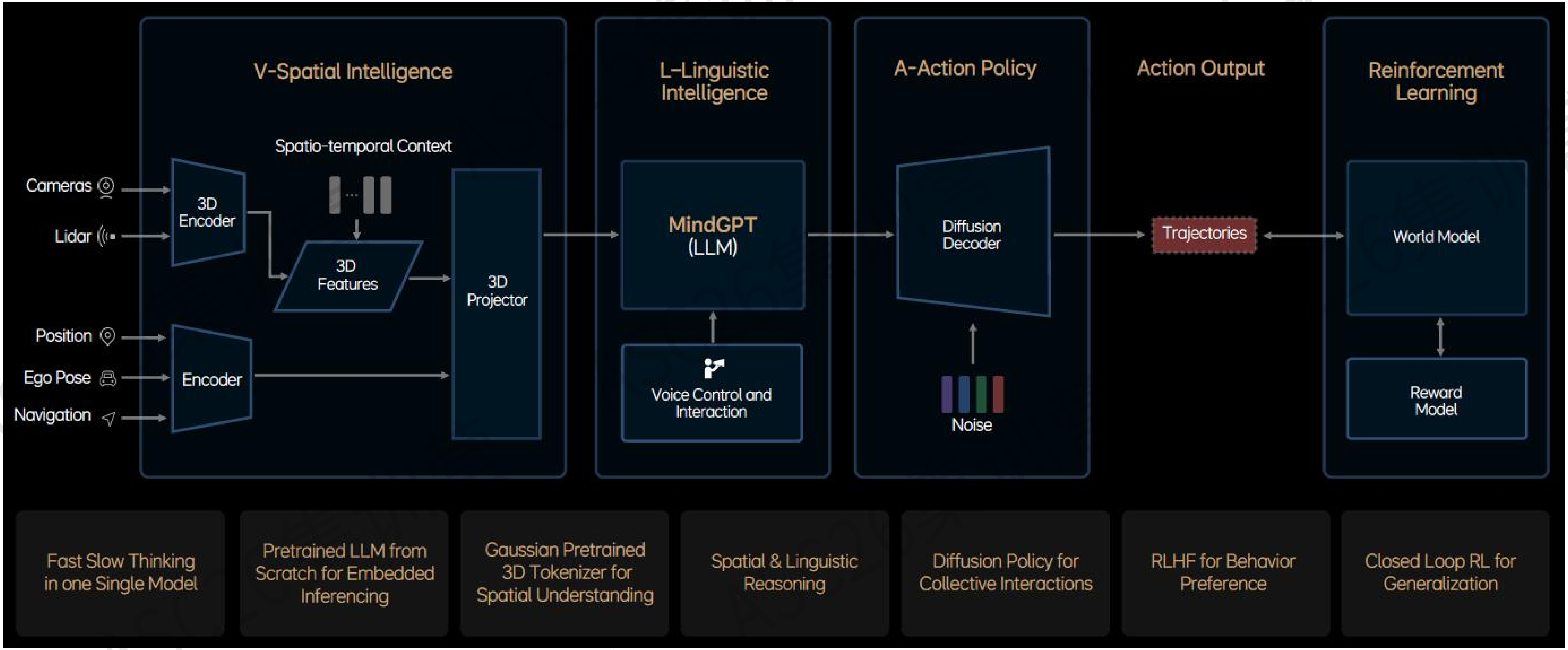
World Model & VLA



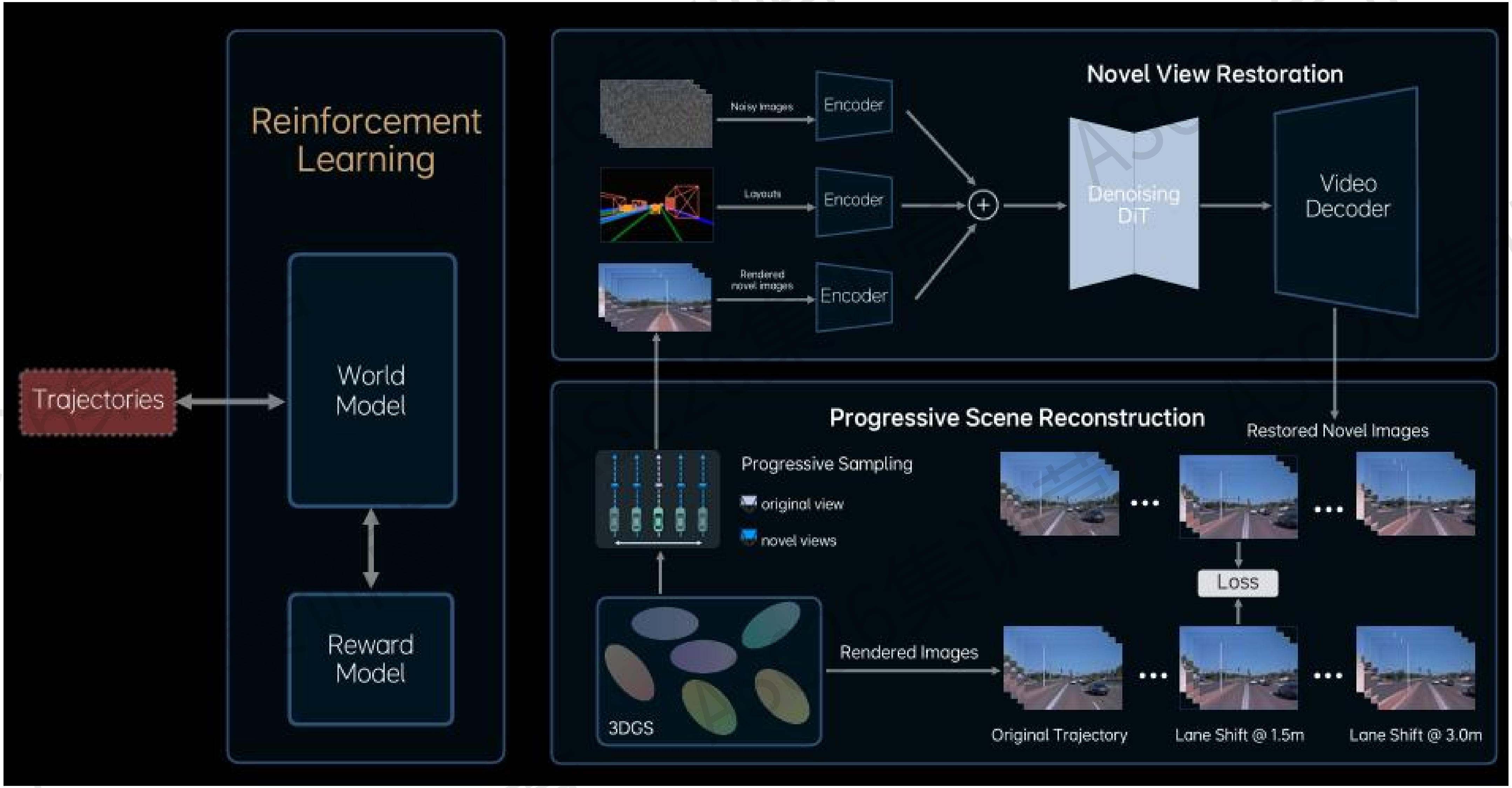
World Model & VLA



World Model & VLA



World Model & VLA



World Model : Cosmos

Data Generation

SIGGRAPH 2025-8

Cosmos Predict

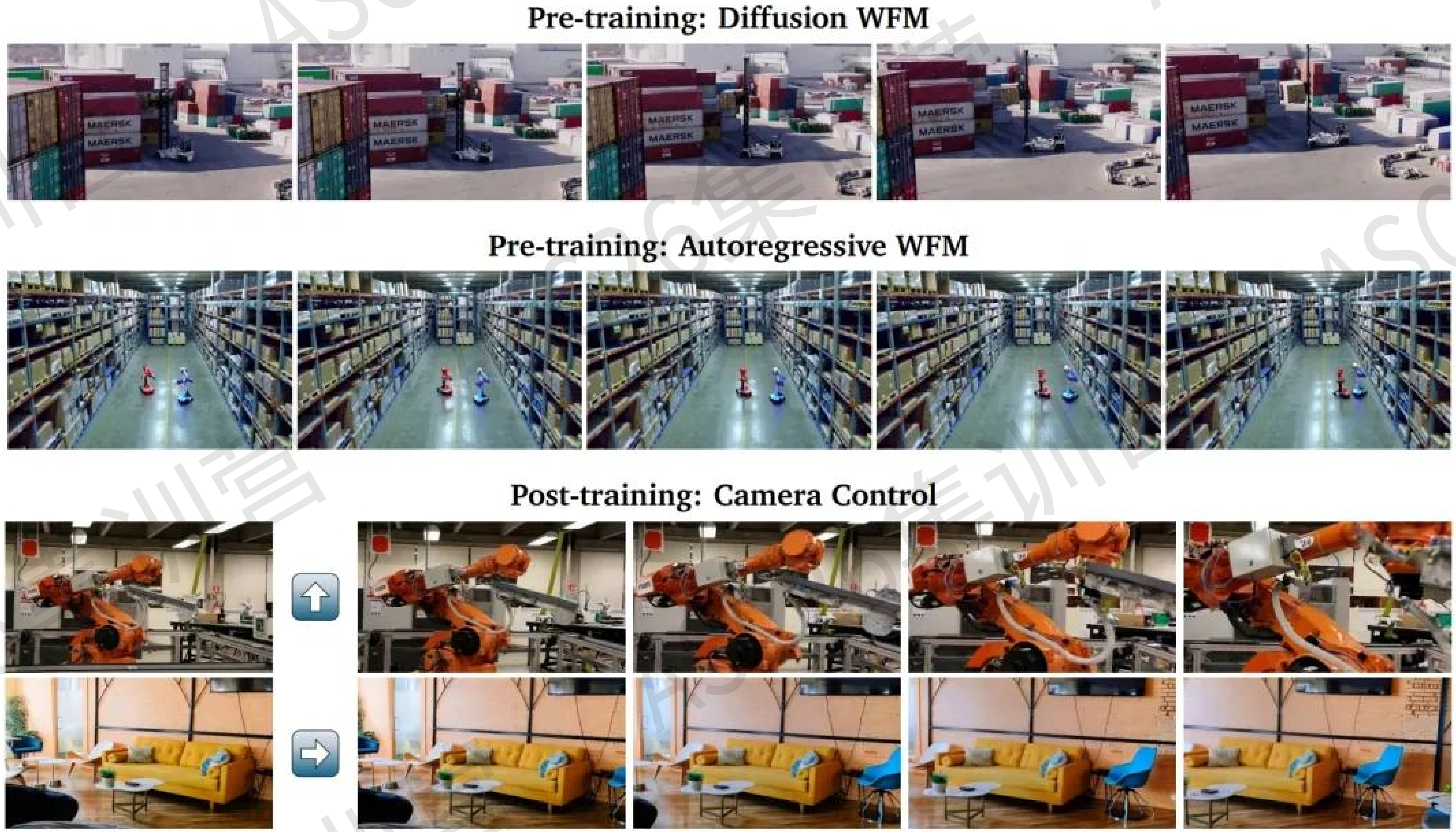
Cosmos is a world model platform featuring a series of open-source, open-weight video world models with parameters ranging from 4B to 14B. The purpose of these models is clear: to generate massive amounts of photorealistic, physics-based synthetic data for AI systems operating in the physical world—such as robots and autonomous vehicles—thereby addressing the severe data shortag

Model Family

Model name	Description	Try it out
Cosmos-1.0-Diffusion-7B-Text2World	Text to visual world generation	Inference
Cosmos-1.0-Diffusion-14B-Text2World	Text to visual world generation	Inference
Cosmos-1.0-Diffusion-7B-Video2World	Video + Text based future visual world generation	Inference
Cosmos-1.0-Diffusion-14B-Video2World	Video + Text based future visual world generation	Inference
Cosmos-1.0-Autoregressive-4B	Future visual world generation	Inference
Cosmos-1.0-Autoregressive-12B	Future visual world generation	Inference
Cosmos-1.0-Autoregressive-5B-Video2World	Video + Text based future visual world generation	Inference
Cosmos-1.0-Autoregressive-13B-Video2World	Video + Text based future visual world generation	Inference
Cosmos-1.0-Guardrail	Guardrail contains pre-Guard and post-Guard for safe use	Embedded in model inference scripts

Cosmos Predict

A pre-training and post-training paradigm is proposed, dividing WFM into pre-training WFM and post-training WFM. To build the pre-training WFM, they leverage large-scale video training datasets to expose the model to diverse visual experiences, transforming it into a generalist model. For the post-training WFM, they fine-tune the pre-trained WFM using datasets collected from specific physical AI environments, thereby creating specialized WFMs tailored for targeted specialized p



Cosmos Predict

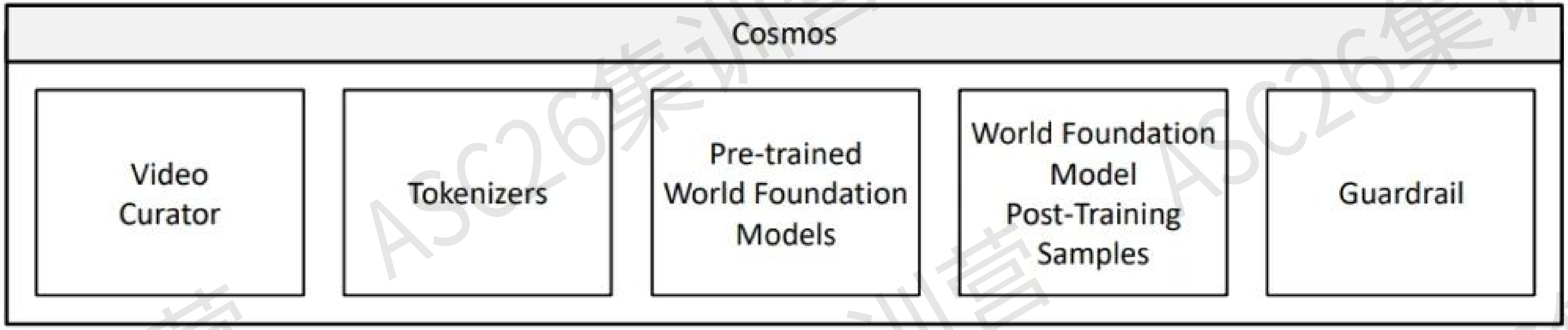


Figure 4: Cosmos World Foundation Model Platform consists of several major components: video curator, video tokenizer, pre-trained world foundation model, world foundation model post-training samples, and guardrail.

Video Curator: Extracted approximately 100 million video clips from a 20 million-hours video collection, with clip durations ranging from 2 to 60 seconds. For each clip, VLM generates video descriptions **every 256 frames**.

Video Tokenization: Developed a series of video tokenizers with varying compression ratios. The token computation for the current frame does not rely on future observations.

WFM Pre-training: Utilized diffusion models and autoregressive models for training.

World Model Post-training: Applied the pre-trained WFM to multiple downstream physical AI applications.

Guardrails: To ensure the safe deployment of the developed world foundation models, a guardrail system was implemented to block harmful inputs and outputs.

Cosmos Predict

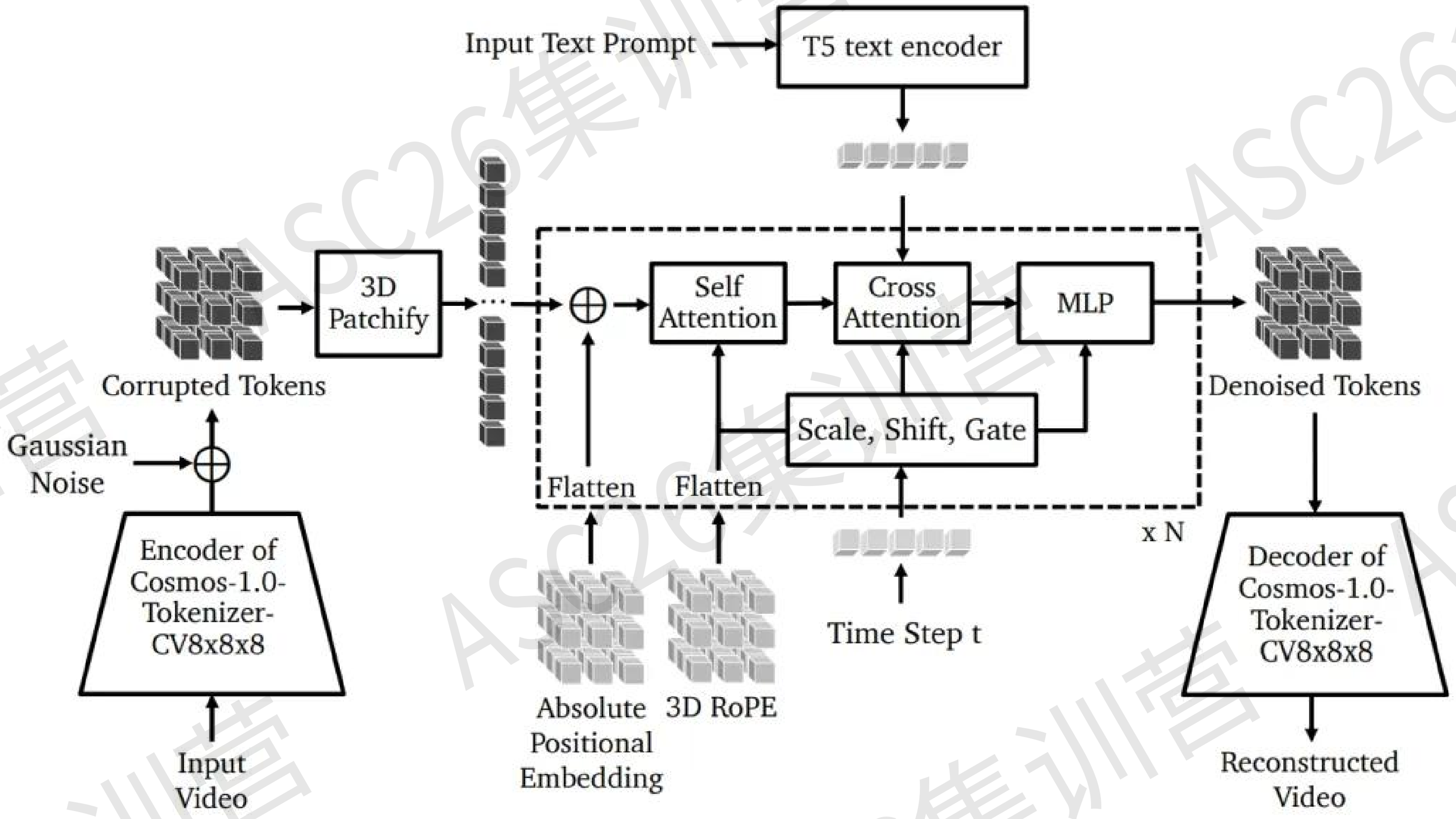


Figure 11: **Overall architecture of Cosmos-1.0-Diffusion World Foundation Model**. The model processes an input video through the encoder of the Cosmos-1.0-Tokenizer-CV8x8x8 to obtain latent representations, which are subsequently perturbed with Gaussian noise. These representations are then transformed using a 3D patchification process. In the latent space, the architecture applies repeated blocks of self-attention, cross-attention (integrating input text), and feed-forward MLP layers, modulated by adaptive layer normalization (scale, shift, gate) for a given time step t. The decoder of Cosmos-1.0-Tokenizer-CV8x8x8 reconstructs the final video output from the refined latent representation.

Cosmos Predict

Table 11: Configuration details of Cosmos-1.0-Diffusion models.

Configuration	7B-Text2World	14B-Text2World	7B-Video2World	14B-Video2World
Number of Layers	28	36	28	36
Model Dimension	4,096	5,120	4,096	5,120
FFN Hidden Dimension	16,384	20,480	16,384	20,480
AdaLN-LoRA Dimension	256	256	256	256
Number of Attention Heads	32	40	32	40
Number of Key / Value Heads	32	40	32	40
MLP Activation	GELU			
Positional Embedding	Hybrid positional embedding			
Conditional Information	Text; FPS	Text; FPS	Text; FPS; Frames	Text; FPS; Frames
Base Learning Rate	2^{-15}	2^{-16}	2^{-15}	2^{-16}
Weight decay	0.1	0.2	0.1	0.2
Learning Rate Warmup	Linear scheduler with 2,500 iterations			
AdamW momentum and ϵ	$\beta_1, \beta_2 = 0.9, 0.99; \epsilon = 10^{-10}$			

Cosmos Predict

Table 12: Stages of progressive training and their specifications.

Stage	Resolution	Number of Frames	Context Length	FSDP Size	CP Size
Low-resolution Pre-training	512p (640×512)	57	10,240 [a]	64	2
High-resolution Pre-training	720p (1280×704)	121	56,320 [b]	64	8
High-quality Fine-tuning	720p (1280×704)	121	56,320 [b]	64	8

[a] 10,240 (the context length) is computed as: 640 (width) $\div 8$ (tokenize) $\div 2$ (patchify) $\times 512$ (height) $\div 8$ (tokenize) $\div 2$ (patchify) $\times [(57 - 1) \div 8 + 1]$ (tokenize frames).
[b] 56,320 (the context length) is computed as: 1280 (width) $\div 8$ (tokenize) $\div 2$ (patchify) $\times 704$ (height) $\div 8$ (tokenize) $\div 2$ (patchify) $\times [(121 - 1) \div 8 + 1]$ (tokenize frames).

Cosmos Predict

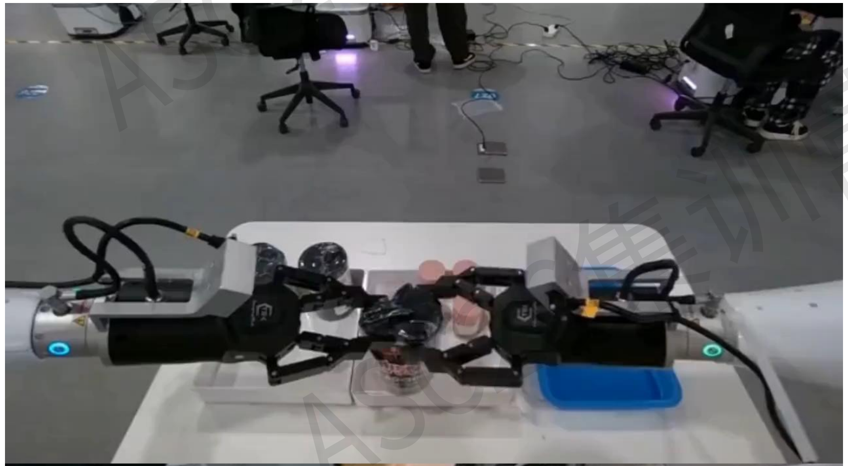
GPU Memory Requirements

The four primary components consuming GPU memory are:

- **Model Parameters**: 10 bytes per parameter. Mixed-precision training stores model parameters in FP32 and BF16 formats, while Exponential Moving Average (EMA) weights are stored in FP32.
- **Gradients**: 2 bytes per parameter. Gradients are stored in BF16.
- **Optimizer States**: 8 bytes per parameter. AdamW (Loshchilov & Hutter, 2019) is used as the optimizer, with its states (first- and second-order moments) stored in FP32.
- **Activations**: (2 × number_of_layers × 15 × seq_len × batch_size × d_model) bytes. Activations are stored in BF16. Selective activation checkpointing (Chen, 2016; Korthikanti, 2023) is implemented to optimize memory usage by recomputing activations for memory-intensive layers (e.g., normalization functions).

- **Example:** A 14B model (e.g., Cosmos-1.0-Diffusion-14B-Text2World) requires approximately **280 GB** for model parameters, gradients, and optimizer states, plus **310 GB** for activations during high-resolution pretraining. Given the 80GB HBM3 memory limit of NVIDIA H100 GPUs, **Fully Sharded Data Parallelism (FSDP)** and **Context Parallelism (CP)** are employed to distribute memory demands across multiple GPUs.

Cosmos Predict



Cosmos Predict

Input Video

Output Video

World Model Challenge

- **Long-term Temporal Consistency:** Achieving long-term temporal consistency and mitigating error accumulation in sequential prediction is a core modeling challenge.

- **Lack of Physically Consistent Evaluation Metrics:** There is an urgent need to develop metrics for evaluating the physical consistency and causality of models, rather than focusing solely on pixel fidelity.

- **Efficiency-Performance Trade-off:** A balance needs to be struck between model performance and computational efficiency required for real-time control on physical devices.

- **Data Scarcity and Unification:** There is a lack of unified, large-scale datasets for embodied AI.

- **Model Interpretability and Robustness:** Existing model-based approaches, such as the Dreamer series, have limitations in interpretability, robustness, and reliability when operating in real-world environments.

Thanks