Vision-Language-Action Models for Embodied AI

A Deep Dive into the 2024 Survey

Mohibkhan Pathan
CMPE 258 - Deep Learning
Professor Vijay Eranti

Table of contents

01 What is Embodied AI?

02 Why Do We Need VLA Models?

03 What Are VLA Models?

04 The Three Main Parts of VLA Models

05 Key Components

06 Low-Level Control

07 High-Level Task Planners

08 Training Data and Benchmarks

09 Challenges

10 The Future of VLAs

11 References

What is Embodied AI?

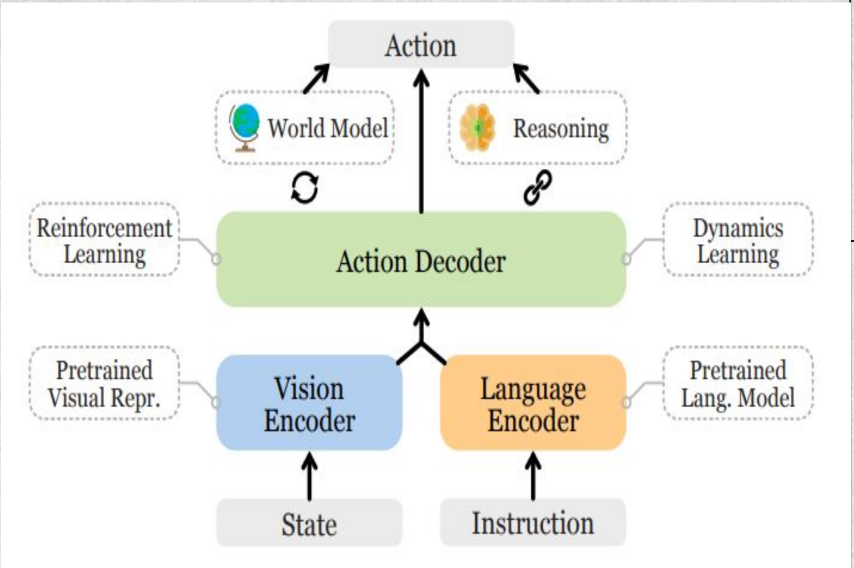
- AI that lives in the real world, not just on a screen
- Can see, understand, and take actions
- Example: A robot that follows the command “bring me a cup”
- It combines vision, language, and movement
- Not like ChatGPT or CLIP – they don’t interact with the physical world

Why Do We Need VLA Models?

- Real-world tasks are complex and multi-step
- Robots must understand what to do, what they see, and how to act
- One model for just vision or just language is not enough
- VLA models let robots follow natural commands like humans
- Helps in homes, hospitals, factories, and more

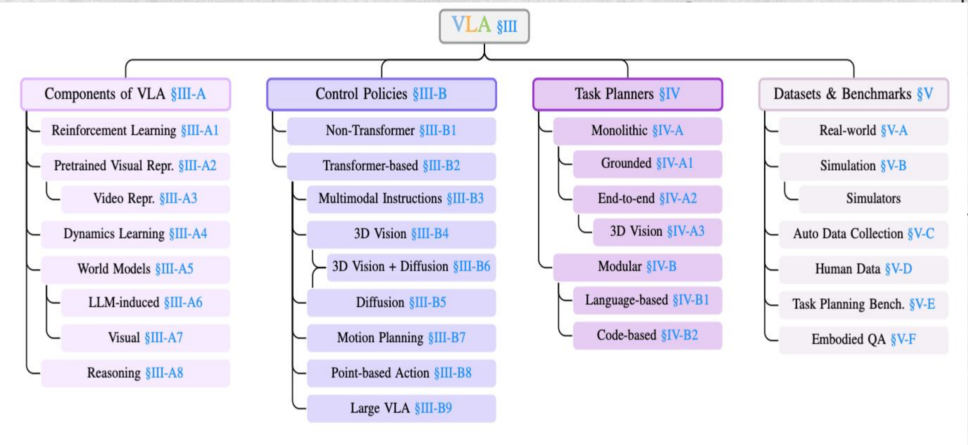
What are VLA Models?

- Real-world tasks are complex and multi-step
- Robots must understand what to do, what they see, and how to act
- One model for just vision or just language is not enough
- VLA models let robots follow natural commands like humans
- Helps in homes, hospitals, factories, and more



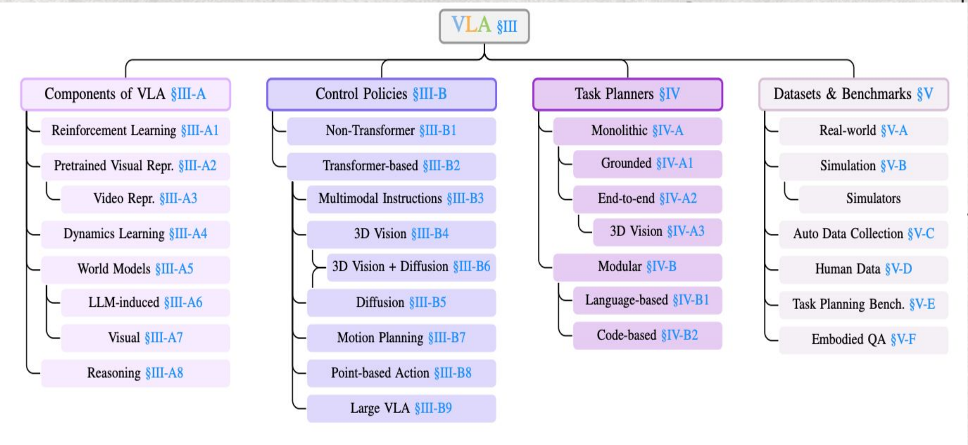
The Three Main Parts of VLA Models

- VLA models have 3 main parts:
    1. **Components** – visual/language encoders and world models
    2. **Low-Level Control** – small step actions (e.g. move, pick)
    3. **High-Level Planners** – break big tasks into small steps
- This structure helps models plan and act better
- Like a team: planner = brain, controller = hands



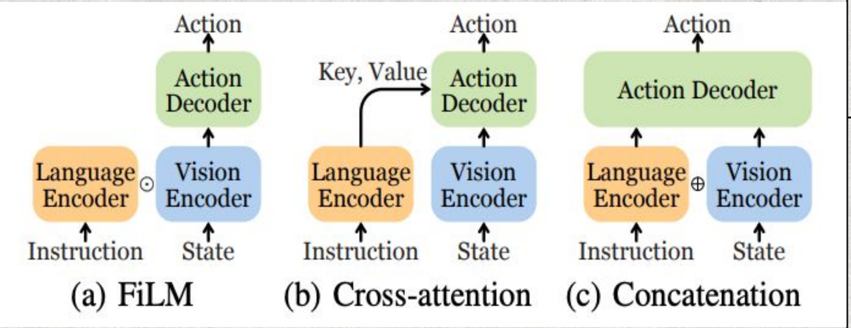
Key Components

- Vision Encoder – helps robot "see" (e.g. CLIP, R3M, MVP)
- Language Encoder – understands commands (e.g. BERT, GPT)
- Dynamics Model – learns how actions change the world
- World Model – predicts what will happen next (like a mini-simulator)
- These parts work together to help the robot think before it moves



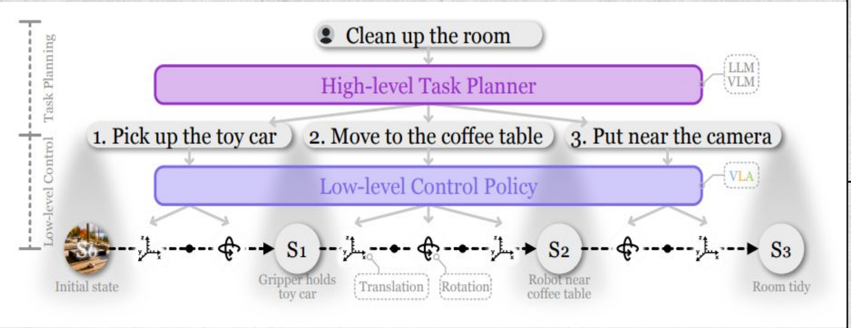
Low-Level Control

- Low-level control = small actions like pick, move, turn
- Uses info from camera + instruction to act in real-time
- Common methods:
  - FiLM – adjusts vision using language
  - Cross-Attention – connects vision + language deeply
  - Concatenation – joins both inputs together
- Models: CLIPort, BC-Z, RT-1, UniPi
- Some use transformers or even learn from videos



High-Level Task Planners

- Planner breaks long tasks into smaller actions
- (e.g. “clean room” → pick up toy → wipe table)
- Two common types:
    - Language-based: LLM writes out steps in text
    - Code-based: LLM creates commands using functions like pick() or move()
- Helps robot know what to do next
- Famous examples: SayCan, InnerMonologue, ProgPrompt



Training Data and Benchmarks

- Real robot data is hard to collect and expensive
- Simulators are used for faster and safer training (e.g. Habitat, AI2-THOR)
- Some models learn from human videos or internet data
- Benchmarks help test models:
  - EmbodiedQA: ask + explore
  - RLBench: robot manipulation
  - EgoPlan / PlanBench: test planning skills
- Important to compare models fairly

Name	Scenes /Rooms	Objects /Cat	UI	Physics Engine	Task	Observation	Action	Agent	Description	Related
Gibson [190]	572/-	-	-	Pybullet	Navi	RGB, D, N, S	-	-	Navi only	-
iGibson [189], [191], [192]	15/108	152/5	Mouse, VR	Pybullet	Navi, Mani	RGB, D, S, N, Flow, LiDAR	Force	TurtleBot v2, LoCoBot, etc.	VR, Continuous Extended States. Versions: iGibson 0.5, 1.0, 2.0	Benchmarks: BEHAVIOR-100 [188], BEHAVIOR-1K [193]
SAPIEN [194]	-	2346/-	Code	PhysX	Navi, Mani	RGB, D, S	Force	Franka	Articulation, Ray Tracing	VoxPoser. Benchmarks: SIMPLER [195]
AI2-THOR [196]	-/120	118/118	Mouse	Unity	Navi, Mani	RGB, D, S, A	Force, PD	ManipulaTHOR, LoCoBot, etc.	Object States, Task Planning. Versions: [197], [198]	Benchmarks: ALFRED, RoPOR [199]
VirtualHome [200]	7/-	-/509	Lang	Unity	Navi, Mani	RGB, D, S	Force, PD	Human	Object States, Task Planning	LID, Translated⟨LM⟩, ProgPrompt
TDW [201]	15/120	112/50	VR	Unity, Flex	Navi, Mani	RGB, D, S, A	Force	Fetch, Sawyer, Baxter	Audio, Fluids	-
RLBench [102]	1/-	28/28	Code	Bullet	Mani	RGB, D, S	Force	Franka	Tiered Task Difficulty	Hiveformer, PerAct
Meta-World [202]	1/-	80/7	Code	MuJoCo	Mani	Pose	Force	Sawyer	Meta-RL	R3M, VC-1, Vi-PRoM, EmbodiedGPT
CALVIN [203]	4/-	7/5	-	Pybullet	Mani	RGB, D	Force	Franka	Long-horizon Lang-cond tasks	GR-1, HULC, RoboFlamingo
Franka Kitchen [204]	1/-	10/6	VR	MuJoCo	Mani	Pose	Force	Franka	Extended by R3M with RGB	R3M, Voltron, Vi-PRoM, Diffusion Policy, EmbodiedGPT
Habitat [205], [206]	Matterport + Gibson		Mouse	Bullet	Navi	RGB, D, S, A	Force	Fetch, Franka, AlienGO	Fast, Navi only. Versions: Rearrangement [207], Habitat 2.0	VC-1, PACT; OVMM [208]
ALFRED [209]	-/120	84/84	-	Unity	Navi, Mani	RGB, D, S	PD	Human	Diverse long-horizon tasks	$(SL)^3$, LLM-Planner
DMC [210]	1/-	4/4	Code	MuJoCo	Control	RGB, D	Force	-	Continuous RL	VC-1, SMART
OpenAI Gym [211]	1/-	4/4	Code	MuJoCo	Control	RGB	Force	-	Single agent RL environments	-
Genesis [212]	(Rigid, deformable, liquid, etc.)		Code	(Proprietary)	Navi, Mani	RGB, D, S, N	Force	Franka, Unitree, etc.	High-speed comprehensive physics simulation	-

Challenges

- Real data is hard to get – robot demos take time
- Models are slow – need to act faster in real life
- System is complex – many parts must work together
- Struggle with new tasks – not good at generalizing
- No standard tests – hard to compare different models
- Safety is important – robots must be trusted by people

The Future of VLA Models

- Smarter planning with better world models
- Faster and smaller models for real-time use
- Use in homes, hospitals, factories, and more
- Safer and more human-friendly robot behavior
- Learn from the world just like humans do

Resources

Ma, Y., Song, Z., Zhuang, Y., Hao, J., & King, I. (2024). *A Survey on Vision-Language-Action Models for Embodied AI*. arXiv preprint arXiv:2408.14496. https://arxiv.org/abs/2408.14496