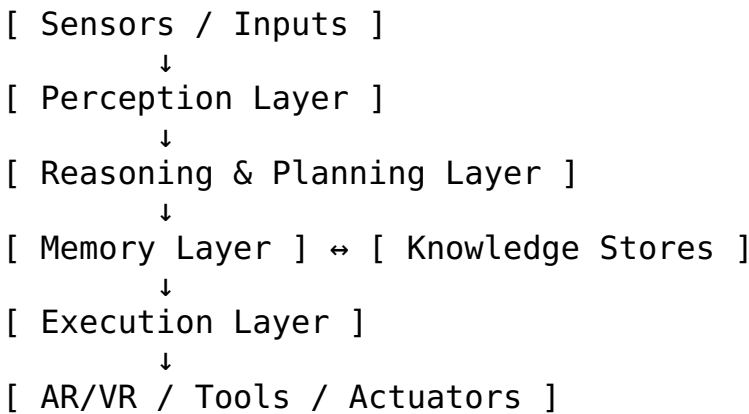


# Architecture & Design Document

## 1. Architectural Overview

The system follows a **Cognitive Agent Architecture** inspired by perception-reasoning-memory-action loops.



## 2. Perception Architecture

### 2.1 Components

- Vision Encoders: CLIP, SAM, Rex-Omni
- OCR: Tesseract, PyMuPDF
- ASR: Whisper, Vosk, Wav2Vec2

### 2.2 Design Considerations

- Latency vs accuracy trade-offs
- Edge vs cloud inference

## 3. Reasoning & Planning Architecture

### 3.1 Planning Frameworks

Framework	Description	Use Case
Sequential Planning	Step-by-step decomposition	Simple tasks
	Plan + act loops	

Framework	Description	Use Case
Interleaved Planning		Dynamic environments
DPPM	Parallel plan generation	Complex tasks

### 3.2 Reflection System

- Actor: Executes plan
- Evaluator: Scores outcome
- Reflector: Updates strategy

---

## 4. Memory Architecture

### 4.1 Short-Term Memory

- Context window
- Session buffers

### 4.2 Long-Term Memory

Type	Technology	Use
Vector DB	FAISS, Qdrant	Semantic recall
Graph DB	Neo4j	Entity relations
SQL DB	PostgreSQL	Logs & states

---

## 5. Execution Architecture

### 5.1 Tool-Based Execution

- Web search APIs
- Code execution sandboxes
- AR rendering engines

### 5.2 Multimodal Action Space

- GUI clicks

- Voice feedback
- AR overlays

---

## 6. Failure Modes & Mitigations

Failure	Mitigation
Hallucination	Rerankers + Reflection
GUI Misgrounding	Vision + DOM fusion
Loops	State tracking + penalties

---

## 7. Example End-to-End Flow

Component	Function	Example
Sensor	Input	Camera sees yellow leaves
DB	Memory	Retrieve plant disease info
LLM	Reasoning	Diagnose nitrogen deficiency
Actuator	Action	Trigger fertilizer valve

---

## 8. Implementation Complexity Analysis

Subsystem	Complexity	Feasibility
Perception	High	Medium
Reasoning	Medium	High
Memory	Medium	High

Subsystem	Complexity	Feasibility
Execution	High	Medium

---

## 9. Future Extensions

- Autonomous robotics
- Digital twins
- Self-improving agents
- Advanced AR/VR interactions