Introduction to AI Agents

Prepared by Ahnaf Hossain Rafi

Analogy: The Chef in a Kitchen

Think of an AI agent as a skilled chef in a busy kitchen. Just like a chef transforms simple ingredients into a delicious meal, an AI agent takes your goals and figures out how to accomplish them step by step.

It's not just following orders—it's thinking, planning, and adapting to get the job done right.

The Goal: A Pizza from a Recipe

The chef starts with a recipe of a pizza to make, which is like the task or command given to an AI Agent. It's a list of what to do, but the chef has to figure out how to make it happen step by step.

Understanding and Planning

The chef reads the recipe and thinks about it: What ingredients do I need? In what order should I chop, mix, and bake? It makes a quick plan, like "First gather tools, then prep veggies, then assemble and cook."

This is the reasoning and planning part for an AI Agent. The agent "thinks" about the goal and breaks it down into a smart plan using its knowledge.

Action and Interaction: Using Kitchen Tools

Now the chef gets to work. It doesn't do everything alone—it use tools like a knife to chop, an oven to bake, or even ask a helper to grab something. The chef directs everything to make the pizza come together.

This shows how an AI Agent acts in its world. It uses "tools" (like searching the internet or writing code) to handle parts of the job, just like the chef uses kitchen gadgets.

Observation and Adaptation: Checking the Taste

While cooking, the chef watches and tastes. Is the dough too sticky? Add more flour. Is it burning? Turn down the heat. It fixes things on the spot to keep everything on track.

This is the observation and adaptation loop for an AI Agent. After acting, it checks the results and changes the plan if needed, like trying a new step to fix a problem.

Achieving the Objective: The Pizza

By planning, using tools, and adjusting as they go, the chef ends up with a perfect pizza that's hot and tasty. It's way better than just throwing ingredients together without thought.

In the same way, an AI Agent combines thinking, acting, and learning from results to handle tricky tasks that a basic AI couldn't do alone. That's what makes it smart and helpful.

AI Agents

An Agent is a system that leverages an AI model to interact with its environment in order to achieve a user-defined objective. It combines reasoning, planning, and the execution of actions (often via external tools) to fulfill tasks.

AI Agents

2 main parts:

1. The Brain (AI Model): The AI model handles reasoning and planning. It decides which Actions to take based on the situation.
2. The Body (Capabilities and Tools): This part represents everything the Agent is equipped to do. The scope of possible actions depends on what the agent has been equipped with. For example, because humans lack wings, they can’t perform the “fly” Action, but they can execute Actions like “walk”, “run” ,“jump”, “grab”, and so on.

Level of Agency

Level	Description	What that’s called	Example pattern
1	Agent output has no impact on program flow	Simple processor	process_llm_output(llm_respo nse)
2	Agent output determines basic control flow	Router	if llm_decision(): path_a() else: path_b()
3	Agent output determines function execution	Tool caller	run_function(llm_chosen_tool, llm_chosen_args)
4	Agent output controls iteration and program continuation	Multi-step Agent	while llm_should_continue(): execute_next_step()
5	One agentic workflow can start another agentic workflow	Multi-Agent	if llm_trigger(): execute_agent()

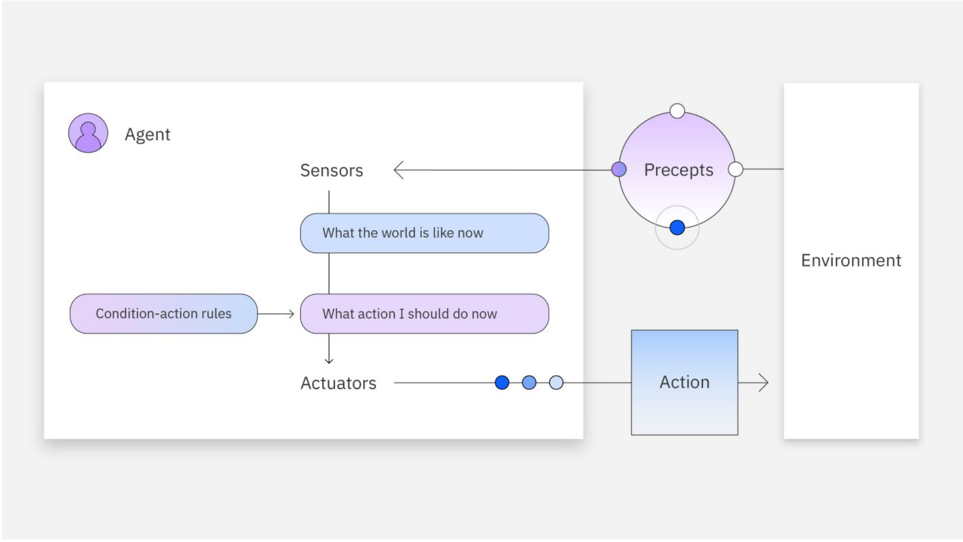
Quiz

Types of AI Agents

- Simple reflex agents
- Model-based reflex agents
- Goal-based agents
- Utility-based agents
- Learning agents

Simple Reflex Agents

- Acts directly on environmental inputs using condition–action rules.
- Senses the environment and responds with fixed, predefined rules, without using memory or future planning.
- Work well in structured, predictable environments with clearly defined rules.
- Can't learn, adapt, or avoid repeating mistakes since they ignore past experiences and future consequences.

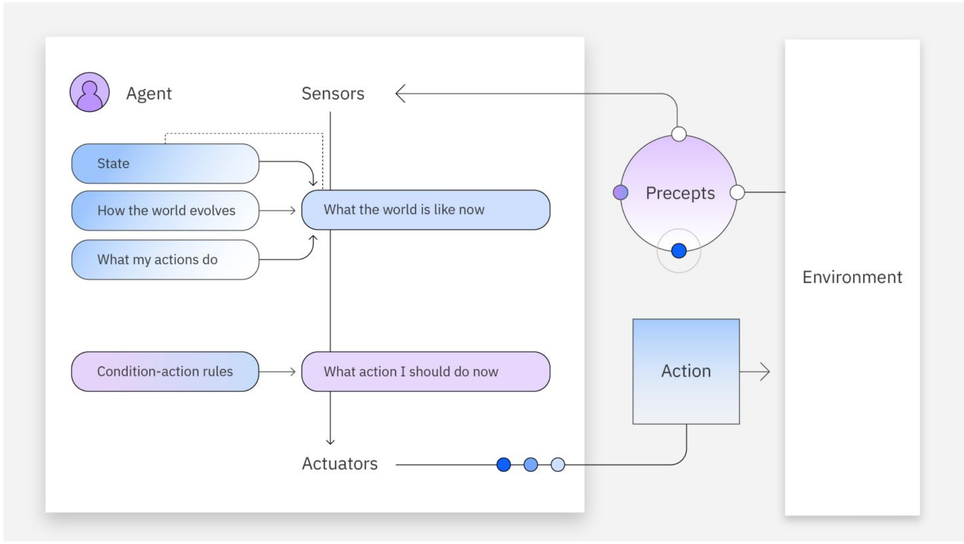


Simple Reflex Agents

- A cleaning robot vacuums whenever its dirt sensor detects dust.
- If dirt detected → start vacuuming. If no dirt → stop.
- Doesn’t remember where it already cleaned; it might keep cleaning the same spot.

Model-based Reflex Agents

- Builds on simple reflex agents by using an internal model of the world alongside condition–action rules.
- Tracks the current state of the environment and considers past interactions to make more informed decisions.
- Work well in partially observable environments, as they can recall context and adapt better than simple reflex agents.
- Lack advanced reasoning and learning abilities for highly dynamic or complex problems.

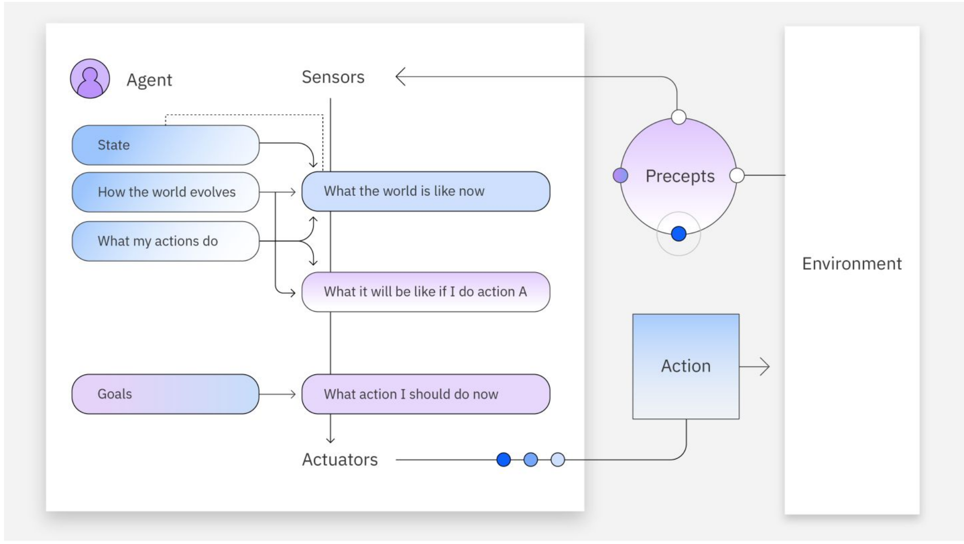


Model-based Reflex Agents

- The robot keeps an internal map of where it has already cleaned.
- If sensor detects dirt AND the map shows this spot hasn't been cleaned → vacuum, else move on.
- Remembers past states, avoids redundant cleaning.

Goal-based Agents

- Improves on simple reflex agents by incorporating goal-oriented decision-making.
- Sets specific goals and uses planning and reasoning to choose actions that move it closer to achieving them.
- Evaluates different possible actions and select the one most likely to help them reach that goal.
- Rely on preprogrammed strategies, limiting adaptability in complex scenarios..

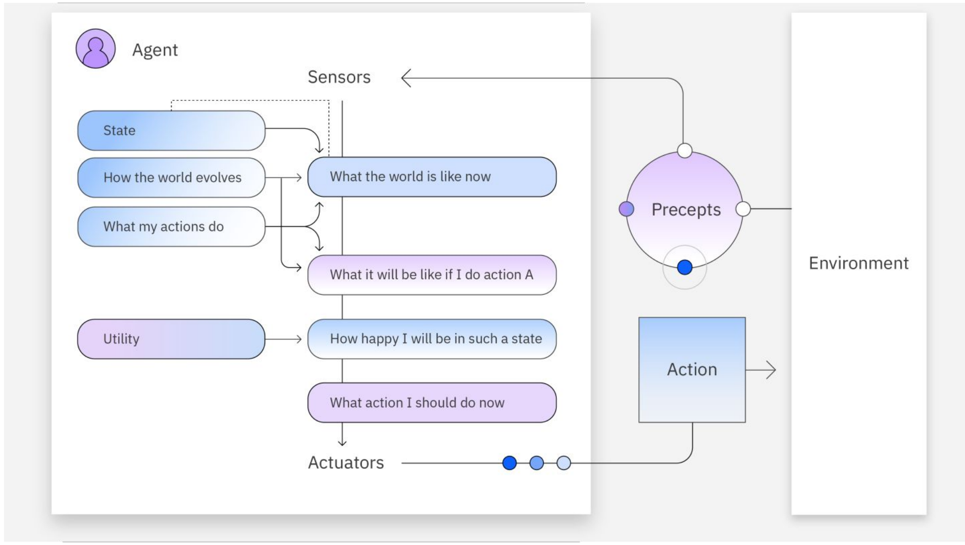


Goal-based Agents

- The robot’s goal is “make the entire room clean.”
- It plans a route to cover all areas of the room efficiently, avoiding unnecessary overlap.
- Doesn’t just react; it reasons about future states to achieve the goal faster.

Utility-based Agents

- Extends goal-based agents by using a utility function to evaluate and choose the most beneficial action.
- Assigns utility values to possible outcomes, balancing multiple goals and tradeoffs to maximize overall benefit.
- Effective in dynamic, complex environments where multiple competing objectives must be considered.
- Designing accurate utility functions is difficult, as it requires weighing many factors and predicting their impacts.

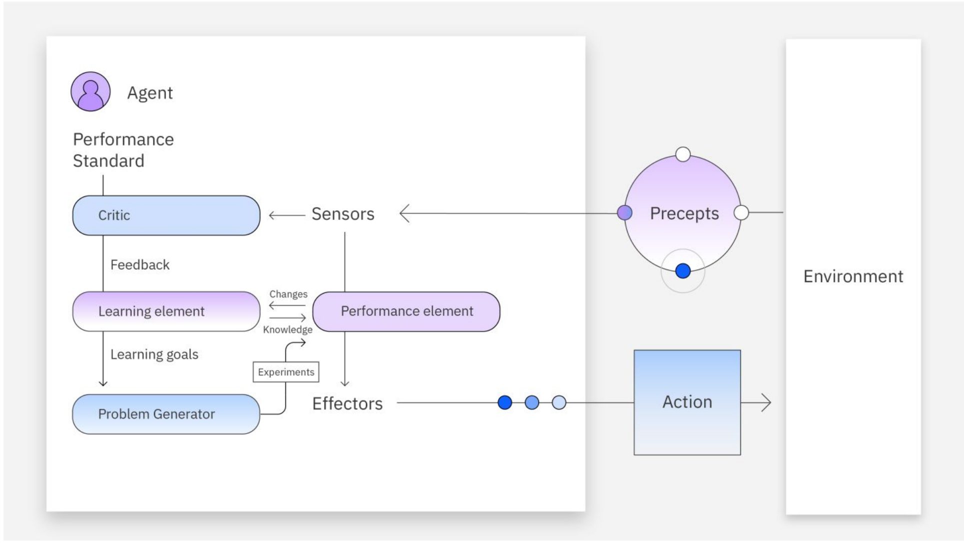

Agent
Sensors
State
How the world evolves
What my actions do
What the world is like now
What it will be like if I do action A
Utility
How happy I will be in such a state
What action I should do now
Actuators
Precepts
Action
Environment

Utility-based Agents

- The robot considers multiple factors: cleanliness level, battery life, and noise.
- It chooses actions that maximize overall utility — e.g., cleans high-traffic areas first, goes to recharge when battery is low, avoids cleaning when people are sleeping.
- Balances tradeoffs instead of blindly pursuing a single goal.

Learning Agents

- Adapts and improves over time by updating its behavior based on experiences and feedback, unlike agents that rely only on fixed rules or models.
- Highly flexible and capable of handling complex, ever-changing environments.
- 4 components:
    - Performance element
    - Learning element
    - Critic
    - Problem generator



Learning Agents

- The robot learns user habits and room layouts over time.
- Notices that the living room gets dirty in the evening and optimizes cleaning schedules accordingly. Learns the most efficient cleaning paths from past runs.
- Continuously adapts to dynamic environments and user preferences.

Quiz

Thank you! Questions?