Artificial Intelligent Agent

by
Assist. Prof. Dr.
Asmaa Abdul Razzaq

An agent is anything that can be viewed as perceiving its **environment** through **sensors** and acting upon that environment through **actuators**.

1. **Sensors:** Eyes (vision), ears (hearing), skin (touch), tongue (Gustation), nose (olfaction), neuromuscular system (proprioception).
2. **Percepts:** At the lowest level – electrical signals, After preprocessing – objects in the visual field (location, textures, colors,), and auditory streams (pitch, loudness, direction).
3. **Actuators:** limbs, digits, eyes, tongue.
4. **Actions:** lift a finger, turn left, walk, run, carry an object,

Vacuum Cleaner World

Percepts: location and contents, e.g. [A, Dirty]
Actions: Left, Right, Suck, NoOp

Vacuum Agent PEAS

1. **Performance Measure**: minimize energy consumption, maximize dirt pick up. Making this precise: one point for each clean square over lifetime of 1000 steps.
2. **Environment**: two squares, dirt distribution unknown, assume actions are deterministic and environment is static (clean squares stay clean)
3. **Actuators**: Left, Right, Suck, NoOp
4. **Sensors**: agent can perceive it's location and whether location is dirty.

Percept sequence	Action
[A, *Clean*]	*Right*
[A, *Dirty*]	*Suck*
[B, *Clean*]	*Left*
[B, *Dirty*]	*Suck*
[A, *Clean*], [A, *Clean*]	*Right*
[A, *Clean*], [A, *Dirty*]	*Suck*
⋮	⋮
[A, *Clean*], [A, *Clean*], [A, *Clean*]	*Right*
[A, *Clean*], [A, *Clean*], [A, *Dirty*]	*Suck*
⋮	⋮

Figure 2.3 Partial tabulation of a simple agent function for the vacuum-cleaner world

Automated taxi driving system

1. **Performance Measure**: Maintain safety, reach destination, maximize profits (fuel, tire wear), obey laws, provide passenger comfort, …
2. **Environment**: U.S. urban streets, freeways, traffic, pedestrians, weather, customers, …
3. **Actuators**: Steer, accelerate, brake, horn, speak/display, …
4. **Sensors**: Video, sonar, speedometer, odometer, engine sensors, keyboard input, microphone, GPS,

Agent Type	Performance Measure	Environment	Actuators	Sensors
Taxi driver	Safe, fast, legal, comfortable trip, maximize profits	Roads, other traffic, pedestrians, customers	Steering, accelerator, brake, signal, horn, display	Cameras, sonar, speedometer, GPS, odometer, accelerometer, engine sensors, keyboard

Figure 2.4 PEAS description of the task environment for an automated taxi.

Autonomy : a property of agents whose behavior is determined by their own experience rather than solely by their initial programming. Therefore, a system is not autonomous if it is guided by its designer according to a priori decisions.

To survive, agents must have:

1. Enough built-in knowledge to survive.
2. The ability to learn

A **rational agent** is one that does the right thing—conceptually speaking, every entry in the table for the agent function is filled out correctly.

What is rational depends on:

1. **P**erformance measure - The performance measure that defines the criterion of success
2. **E**nvironment - The agents prior knowledge of the environment
3. **A**ctuators - The actions that the agent can perform
4. **S**ensors - The agent's percept sequence to date

We'll call all this the Task Environment (**PEAS**)

Properties of Environments:

Environment Types (cont.)

- **Static** (vs. dynamic):
  - The environment is unchanged while an agent is deliberating (the environment is semi-dynamic if the environment itself does not change with the passage of time but the agent's performance score does).
- **Discrete** (vs. continuous):
  - A limited number of distinct, clearly defined percepts and actions.
- **Single agent** (vs. multi-agent):
  - An agent operating by itself in an environment.

- **Fully observable** (vs. partially observable):
  - An agent's sensors give it access to the complete state of the environment at each point in time.
- **Deterministic** (vs. stochastic):
  - The next state of the environment is completely determined by the current state and the action executed by the agent. (If the environment is deterministic except for the actions of other agents, then the environment is strategic).
- **Episodic** (vs. sequential):
  - The agent's experience is divided into atomic "episodes" (each episode consists of the agent perceiving and then performing a single action), and the choice of action in each episode depends only on the episode itself.

Task Environment	Observable	Agents	Deterministic	Episodic	Static	Discrete
Crossword puzzle	Fully	Single	Deterministic	Sequential	Static	Discrete
Chess with a clock	Fully	Multi	Deterministic	Sequential	Semi	Discrete
Poker	Partially	Multi	Stochastic	Sequential	Static	Discrete
Backgammon	Fully	Multi	Stochastic	Sequential	Static	Discrete
Taxi driving	Partially	Multi	Stochastic	Sequential	Dynamic	Continuous
Medical diagnosis	Partially	Single	Stochastic	Sequential	Dynamic	Continuous
Image analysis	Fully	Single	Deterministic	Episodic	Semi	Continuous
Part-picking robot	Partially	Single	Stochastic	Episodic	Dynamic	Continuous
Refinery controller	Partially	Single	Stochastic	Sequential	Dynamic	Continuous
Interactive English tutor	Partially	Multi	Stochastic	Sequential	Dynamic	Discrete

Figure 2.6 Examples of task environments and their characteristics.

THE STRUCTURE OF AGENTS

Agent function: a function that specifies the agent's action in response to every possible Percept sequence.

function TABLE-DRIVEN-AGENT(*percept*) **returns** an action
  persistent: *percepts*, a sequence, initially empty
    table, a table of actions, indexed by percept sequences, initially fully specified

  append *percept* to the end of *percepts*
  action ← LOOKUP(*percepts*, *table*)
  return *action*

Figure 2.7 The TABLE-DRIVEN-AGENT program is invoked for each new percept and returns an action each time. It retains the complete percept sequence in memory.

Agent program: that program which, combined with a machine architecture, implements an agent function. In our simple designs, the program takes a new percept on each invocation and returns an action.

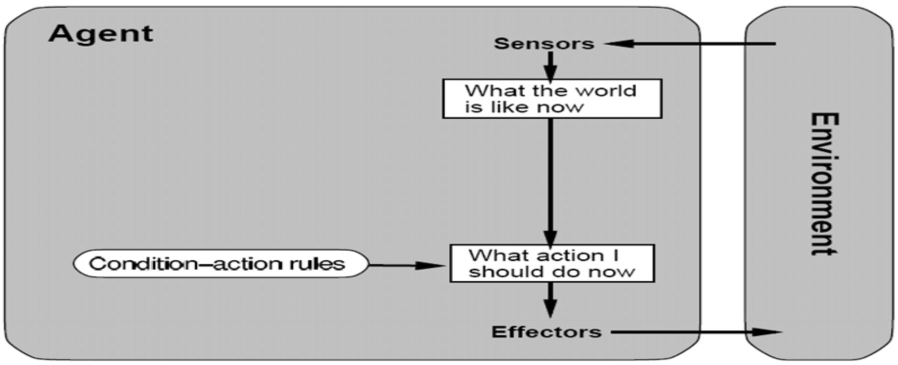
function REFLEX-VACUUM-AGENT([*location*,*status*]) **returns** an action
  if *status* = *Dirty* **then return** *Suck*
  else if *location* = *A* **then return** *Right*
  else if *location* = *B* **then return** *Left*

Figure 2.8 The agent program for a simple reflex agent in the two-state vacuum environment. This program implements the agent function tabulated in Figure 2.3.

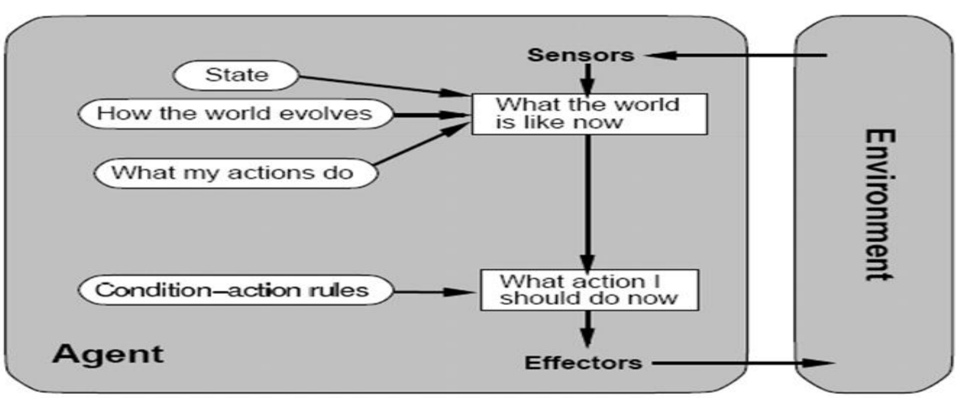
<u>Some agent types</u>

1. **Table-driven agents** : use a percept sequence/action table in memory to find the next action. They are implemented by a (large) **lookup table**.
2. **Simple reflex agents**:

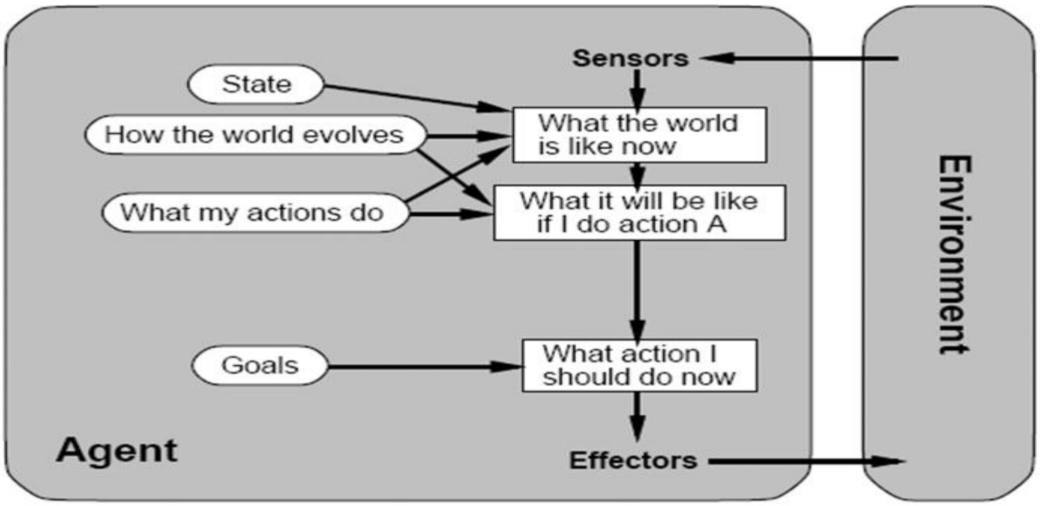
- are based on **condition-action rules(if-then)**,
- Simplest kind of agent.
- Implemented with an appropriate production system.
- They are stateless devices which do not have memory of past world states.



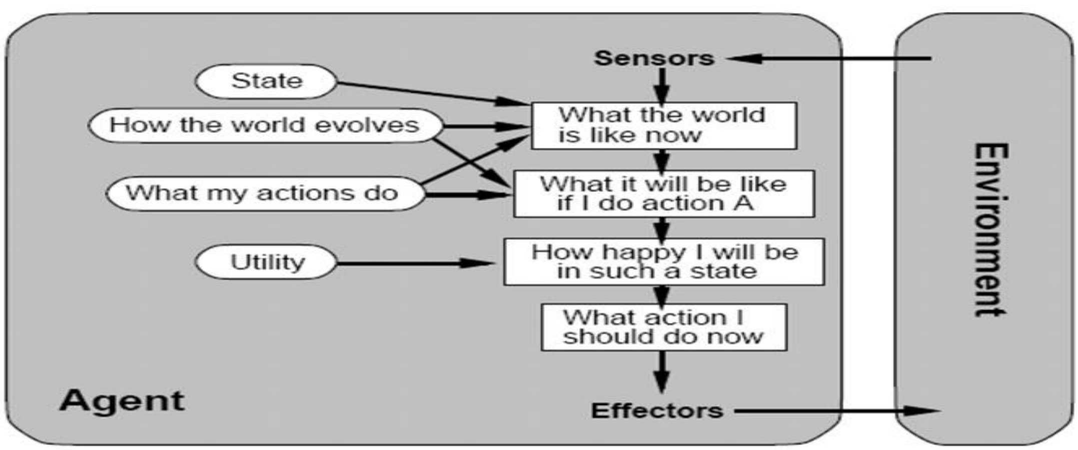
3. **Model-based reflex agents:**
    - Have **internal state**, which is used to keep track of past states of the world.
    - Actions are derived directly from an internal model of the current world state that is updated over time. •
    - Models the state of the world by modeling how the world works •
    - Actions are based on the knowledge about current state (from sensors) and previous states



4. **Goal-based agents:**
   - Are agents that, in addition to state information, have **goal information** that describes desirable situations
   - Agents of this kind take future events into consideration.
   - Involves decision making (not only condition-action rules)

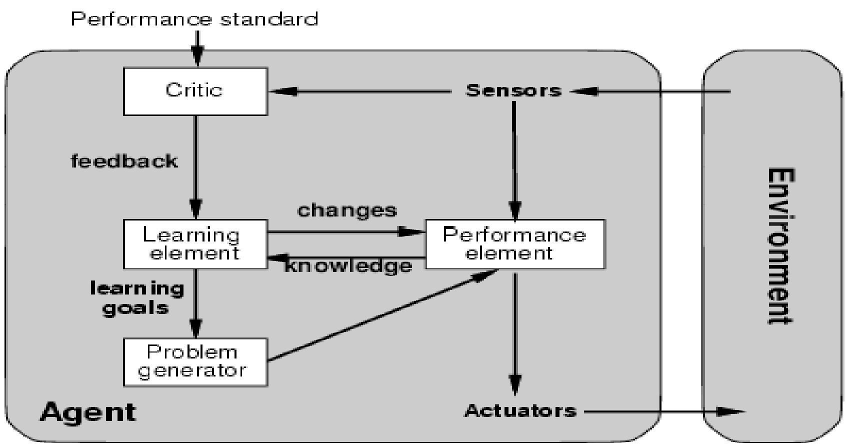


5. **Utility-based agents:**
   - Base their decisions on **classic axiomatic utility theory** in order to act rationally.
   - Some paths to the goal are better (more efficient) than others) - which combination of goals is preferred



Learning agent:

- Behavior improves over time based on its experience
- Monitoring agent's performances over time
- Learn based on preferences of actions (feedback)
- Problem generation – suggesting actions that will lead to new and informative experiences



Performance measure vs. utility function

- A performance measure (typically imposed by the designer) is used to evaluate the behavior of the agent in environment.
- A utility function is used by an agent itself to evaluate how desirable states are. Some paths to the goal are better (more efficient) than others –which path is the best
- The utility function may not be the same as the performance measure
- An agent may have no explicit utility function at all, whereas there is always a performance measure

Summary: Agents

- An agent perceives and acts in an environment, has an architecture, and is implemented by an agent program. Task environment – PEAS (Performance, Environment, Actuators, Sensors)
- An ideal agent always chooses the action which maximizes its expected performance, given its percept sequence so far.
- An autonomous agent uses its own experience rather than built-in knowledge of the environment by the designer.