1.2 **AGENTS**

Agents an0d environments

An **agent** is anything that can be viewed as perceiving its **environment** through **sensors** and SENSOR acting upon that environment through **actuators.** This simple idea is illustrated in Figure 1.2.

- A human agent has eyes, ears, and other organs for sensors and hands, legs, mouth, and other body parts for actuators.
- A robotic agent might have cameras and infrared range finders for sensors and various motors for actuators.
- A software agent receives keystrokes, file contents, and network packets as sensory inputs and acts on the environment by displaying on the screen, writing files, and sending network packets.

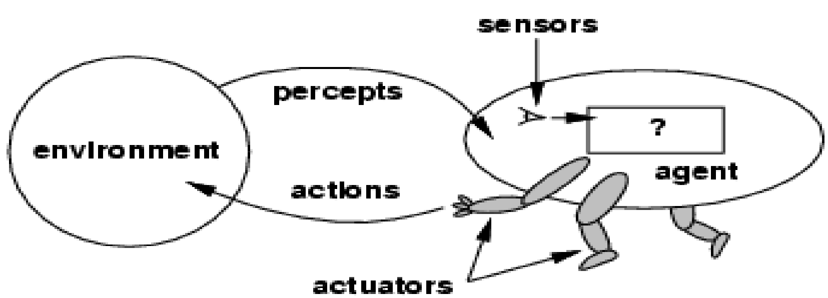


Figure 1.2 Agents interact with environments through sensors and actuators.

Percept

We use the term **percept** to refer to the agent's perceptual inputs at any given instant.

Percept Sequence

An agent's **percept sequence** is the complete history of everything the agent has ever perceived.

Agent function

Mathematically speaking, we say that an agent's behavior is described by the **agent function** that maps any given **percept sequence** to an action.

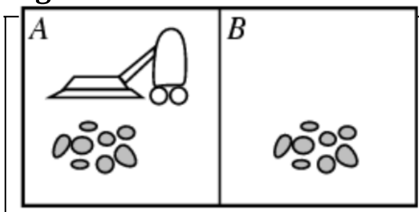
$f : P^* \rightarrow A$

Agent program

Internally, the agent function for an artificial agent will be implemented by an **agent program.** It is important to keep these two ideas distinct. The agent function is an abstract (Existing only in the mind) mathematical description; the agent program is a concrete implementation, running on the agent architecture.

To illustrate these ideas, we will use a very simple example-the vacuum-cleaner world shown in Figure 1.3. This particular world has just two locations: squares A and B. The vacuum agent perceives which square it is in and whether there is dirt in the square. It can choose to move left, move right, suck up the dirt, or do nothing. One very simple agent function is the following: if the current square is dirty, then suck, otherwise move to the other square. A partial tabulation of this agent function is shown in Figure 1.4.

Figure 1.3 A vacuum-cleaner world with just two locations.



Agent function

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
…	

Figure 1.4 Partial tabulation of a simple agent functions for the vacuum-cleaner world shown in Figure 1.3.

agent program

```
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
```

Rational Agent

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. Obviously, doing the right thing is better than doing the wrong thing. The right action is the one that will cause the agent to be most successful.

Performance measures

A performance measure is the **criterion for success** of an agent's behavior. When an agent is plunked down in an environment, it generates a sequence of actions according to the percepts it receives. This sequence of actions causes the environment to go through a sequence of states. If the sequence is desirable, then the agent has performed well.

What is rational at any given time depends on four things:

- The **performance measure** that defines the criterion of success.
- The agent's **prior knowledge** of the environment.
- The **actions** that the agent can perform.
- The agent's **percept sequence** to date.

This leads to a **definition of a rational agent:**

For each possible percept sequence, a rational agent should select an action that is expected to maximize its performance measure, given the evidence provided by the percept sequence and whatever built-in knowledge the agent has.

Omniscience, learning, and autonomy

An **omniscient agent** knows the *actual* outcome of its actions and can act accordingly; but omniscience is impossible in reality.

Doing actions in order to modify future percepts, sometimes called **information gathering** - is an important part of rationality. Our definition requires a rational agent not only to gather information, but also to **learn** as much as possible from what it perceives.

To the extent that an agent relies on the prior knowledge of its designer rather than on its own percepts, we say that the agent lacks autonomy. A rational agent should be **autonomous** - it should learn what it can to compensate for partial or incorrect prior knowledge.

Task environments

We must think about **task environments,** which are essentially the "**problems**" to which rational agents are the "**solutions**."

The rationality of the simple vacuum-cleaner agent, needs specification of

- the **P**erformance measure
- the **E**nvironment
- the agent's **A**ctuators and **S**ensors.

PEAS

We call this the **PEAS** (Performance, Environment, Actuators, and Sensors) description.

In designing an agent, the first step must always be to specify the task environment as fully as possible.

Agent Type	**Performance Measure**	**Environments**	**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, engine sensors, keyboards, accelerometer

Figure 1.5 PEAS description of the task environment for an automated taxi.

Agent Type	Performance Measure	Environment	Actuators	Sensors
Medical diagnosis system	Healthy patient, minimize costs, lawsuits	Patient, hospital, staff	Display questions, tests, diagnoses, treatments, referrals	Keyboard entry of symptoms, findings, patient's answers
Satellite image analysis system	Correct image categorization	Downlink from orbiting satellite	Display categorization of scene	Color pixel arrays
Part-picking robot	Percentage of parts in correct bins	Conveyor belt with parts; bins	Jointed arm and hand	Camera, joint angle sensors
Refinery controller	Maximize purity, yield, safety	Refinery, operators	Valves, pumps, heaters, displays	Temperature, pressure, chemical sensors
Interactive English tutor	Maximize student's score on test	Set of students, testing agency	Display exercises, suggestions, corrections	Keyboard entry

Figure 1.6 Examples of agent types and their PEAS descriptions.

Properties of task environments

- Fully observable vs. partially observable
- Deterministic vs. stochastic
- Episodic vs. sequential
- Static vs. dynamic
- Discrete vs. continuous
- Single agent vs. multiagent

Fully observable vs. **partially observable**

If an agent's sensors give it access to the complete state of the environment at each point in time, then we say that the task environment is fully observable. A task environment is effectively fully observable if the sensors detect all aspects that are *relevant* to the choice of action. An environment might be **partially observable** because of noisy and inaccurate sensors or because parts of the state are simply missing from the sensor data.

Deterministic vs. **stochastic**

If the next state of the environment is completely determined by the current state and the action executed by the agent, then we say the environment is deterministic; other- wise, it is stochastic.

Episodic vs. **sequential**

In an **episodic task environment**, the agent's experience is divided into atomic episodes. Each episode consists of the agent perceiving and then performing a single action. **Crucially, the next episode does not depend on the actions taken in previous episodes.** For example, an agent that has to spot defective parts on an assembly line bases each decision on the current part, regardless of previous decisions;

In **sequential environments**, on the other hand, the current decision could affect all future decisions. Chess and taxi driving are sequential.

Discrete vs. **continuous.**

The discrete/continuous distinction can be applied to the *state* of the environment, to the way *time* is handled, and to the *percepts* and *actions* of the agent. For example, a discrete-state environment such as a chess game has a finite number of distinct states. Chess also has a discrete set of percepts and actions. Taxi driving is a continuous state and continuous time problem: the speed and location of the taxi and of the other vehicles sweep through a range of continuous values and do so smoothly over time. Taxi-driving actions are also continuous (steering angles, etc.).

Single agent vs. **multiagent.**

An agent solving a crossword puzzle by itself is clearly in a single-agent environment, whereas an agent playing chess is in a two-agent environment.

As one might expect, the hardest case is *partially observable, stochastic, sequential, dynamic, continuous,* and *multiagent.*

Figure 1.7 lists the properties of a number of familiar environments.

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

Figure 1.7 Examples of task environments and their characteristics.

Week - 2