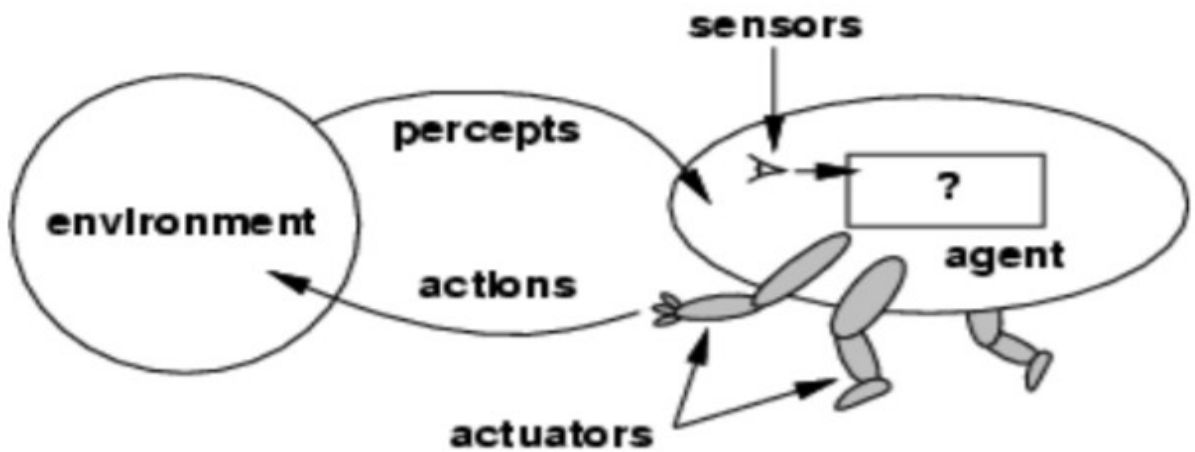
Intelligent Agents

Intelligent agents perceive their environment persists over a prolong time period, adapt to change and create and pursue goals.

Some slides are adopted from University of Illinois CS courses

Agents

- An agent is anything that can be viewed as
  - perceiving its environment through sensors and
  - acting upon that environment through actuators
  - Assumption: Every agent can perceive its own actions (but not always the effects)



Agents

- Human agent:
  - eyes, ears, and other organs for sensors;
  - hands,legs, mouth, and other body parts for actuators

- Robotic agent:
  - cameras and infrared range finders for sensors;
  - various motors for actuators

- A software agent:
  - Keystrokes, file contents, received network packages as sensors
  - Displays on the screen, files, sent network packets as actuators

Rational agents

- An agent should strive to "do the right thing", based on what it can perceive and the actions it can perform.
- The right action is the one that will cause the agent to be most successful
- Performance measure: An objective criterion for success of an agent's behavior
- E.g., performance measure of a vacuum-cleaner agent could be amount of dirt cleaned up, amount of time taken, amount of electricity consumed, amount of noise generated, etc.
- As a general rule, it is better to design performance measures according to what one actually wants in the environment. Rather than according to how one thinks the agent should behave (amount of dirt cleaned vs a clean floor)
- A more suitable measure would reward the agent for having a clean floor

Rationality

- 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

Rational agents

- 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 the agent's built-in knowledge*
- **Performance measure (utility function):** An *objective* criterion for success of an agent's behavior

Specifying the task environment (PEAS)

- PEAS:
  - **P**erformance measure,
  - **E**nvironment,
  - **A**ctuators,
  - **S**ensors
- In designing an agent, the first step must always be to specify the task environment (PEAS) as fully as possible

Specifying the task environment (PEAS)

- PEAS:
  - **P**erformance measure,
  - **E**nvironment,
  - **A**ctuators,
  - **S**ensors
- In designing an agent, the first step must always be to specify the task environment (PEAS) as fully as possible

Another PEAS example: Spam filter

- Performance measure
  - Minimizing false positives, false negatives
- Environment
  - A user's email account, email server
- Actuators
  - Mark as spam, delete, etc.
- Sensors
  - Incoming messages, other information about user's account

PEAS for a medical diagnosis system

- Performance measure: Healthy patient, minimize costs, lawsuits
- Environment: Patient, hospital, staff
- Actuators: Screen display (questions, tests, diagnoses, treatments, referrals)
- Sensors: Keyboard (entry of symptoms, findings, patient's answers)

PEAS for a part-picking robot

- Performance measure: Percentage of parts in correct bins
- Environment: Conveyor belt with parts, bins
- Actuators: Jointed arm and hand
- Sensors: Camera, joint angle sensors

PEAS for a satellite image analysis system

- Performance measure: correct image categorization
- Environment: downlink from orbiting satellite
- Actuators: display categorization of scene
- Sensors: color pixel arrays

PEAS for a refinery controller

- Performance measure: maximize purity, yield, safety
- Environment: refinery, operators
- Actuators: valves, pumps, heaters, displays
- Sensors: temperature, pressure, chemical sensors

PEAS for Interactive English tutor

- Performance measure: Maximize student's score on test
- Environment: Set of students
- Actuators: Screen display (exercises, suggestions, corrections)
- Sensors: Keyboard

Exercise : Specify **PEAS** for an autonomous taxi driver Agent

Performance	
Environment	
Actuators	
Sensors	

Environment types

- 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:

- An environment is fully observable if an agent's sensors give it access to the complete state of the environment at each point in time.
- Fully observable environments are convenient, because the agent need not maintain any internal state to keep track of the world
- 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
- Examples: vacuum cleaner with local dirt sensor, taxi driver

Environment types

Deterministic vs. stochastic:

- The environment is deterministic if the next state of the environment is completely determined by the current state and the action executed by the agent.
- In principle, an agent need not worry about uncertainty in a fully observable, deterministic environment
- If the environment is partially observable then it could appear to be stochastic
- Examples: Vacuum world is deterministic while taxi driver is not
- If the environment is deterministic except for the actions of other agents, then the environment is strategic

Environment types

Episodic vs. sequential:

- In episodic environments, 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.
- Examples: classification tasks

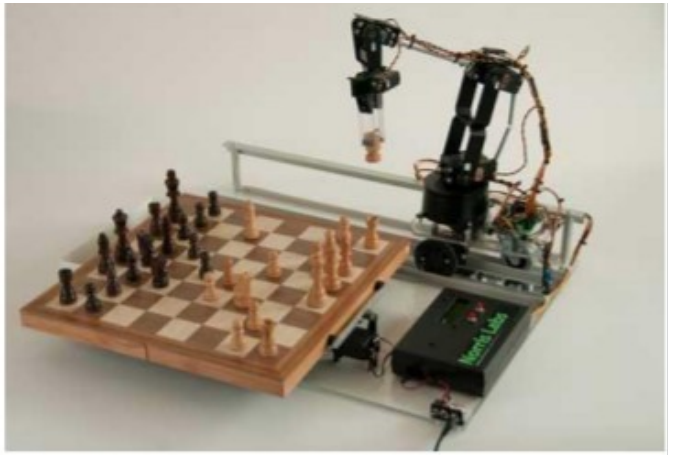
- In sequential environments, the current decision could affect all future decisions
- Examples: chess and taxi driver

Environment types

Static vs. dynamic:

- The environment is unchanged while an agent is deliberating.
- Static environments are easy to deal with because the agent need not keep looking at the world while it is deciding on the action or need it worry about the passage of time
- Dynamic environments continuously ask the agent what it wants to do
- The environment is semi-dynamic if the environment itself does not change with the passage of time but the agent's performance score does
- Examples: taxi driving is dynamic, chess when played with a clock is semi-dynamic, crossword puzzles are static

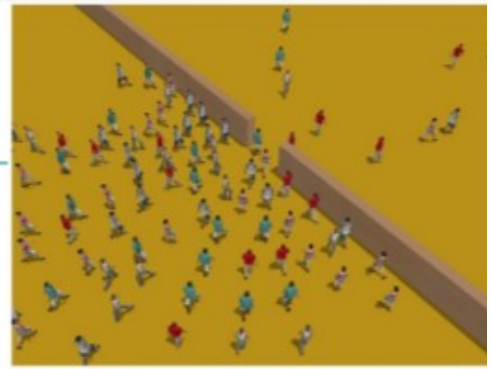
Environment types



Discrete vs. continuous:

- A limited number of distinct, clearly defined states, percepts and actions.
- Examples: Chess has finite number of discrete states, and has discrete set of percepts and actions. Taxi driving has continuous states, and actions

Environment types



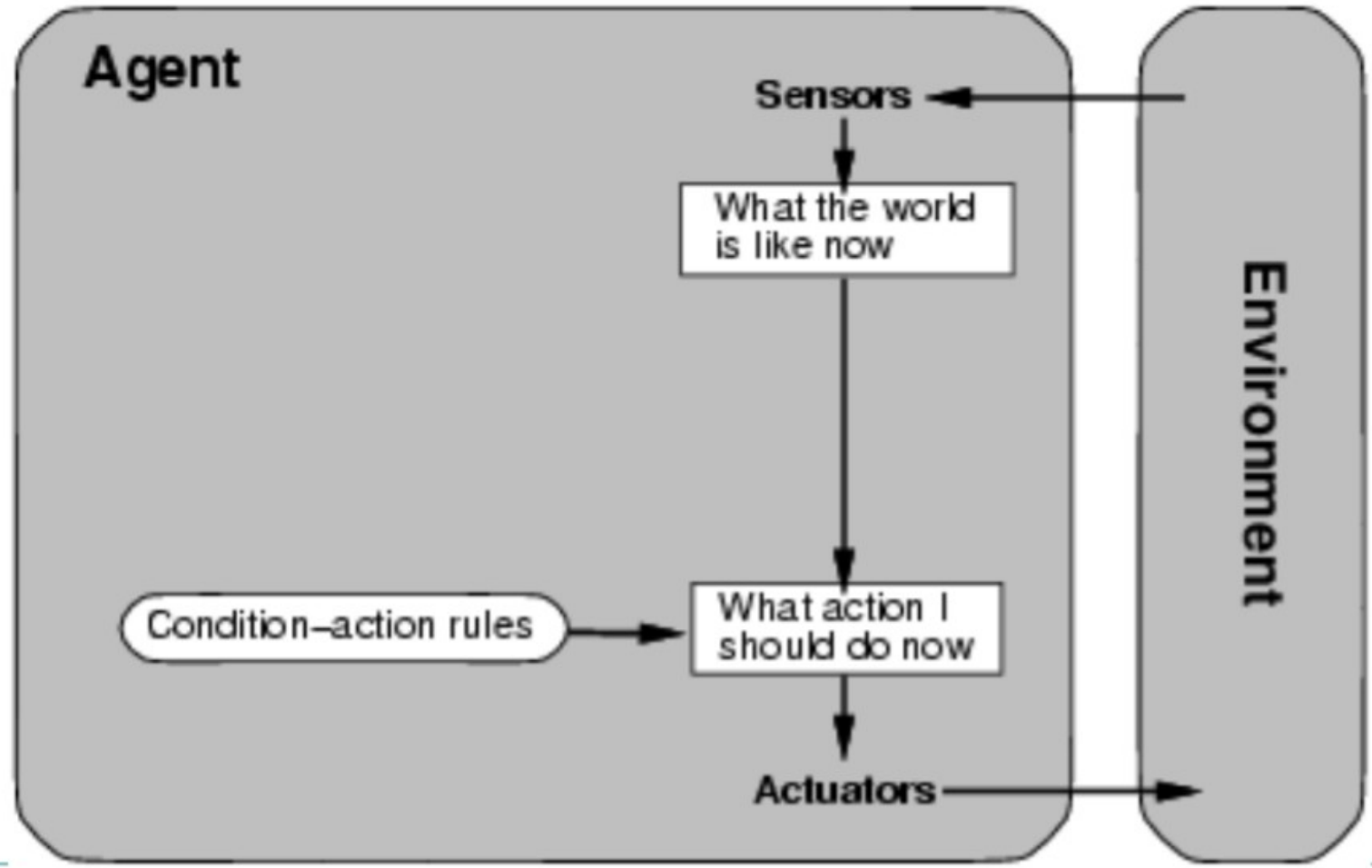
Single agent vs. multiagent:

- An agent operating by itself in an environment is single agent
- Examples: Crossword is a single agent while chess is two-agents

Examples of different environments

	 Word jumble solver	 Chess with a clock	 Scrabble	 Autonomous driving
Observable	Fully	Fully	Partially	Partially
Deterministic	Deterministic	Strategic	Stochastic	Stochastic
Episodic	Episodic	Sequential	Sequential	Sequential
Static	Static	Semidynamic	Static	Dynamic
Discrete	Discrete	Discrete	Discrete	Continuous
Single agent	Single	Multi	Multi	Multi

Simple reflex agents

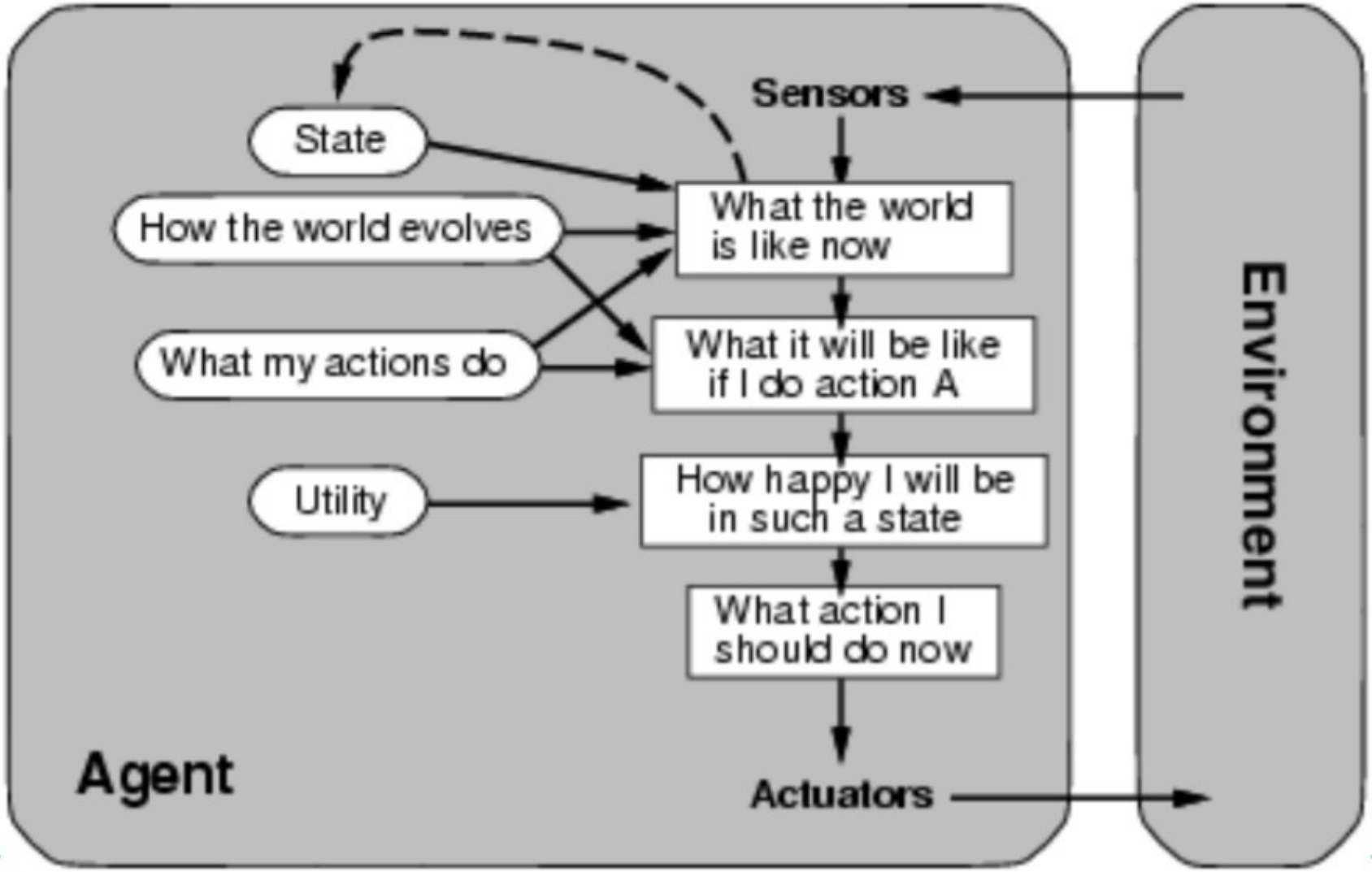


Simple reflex agents

```
function SIMPLE-REFLEX-AGENT(percept) returns an action
static: rules,  a set if condition-action rules
state <-- INTERPRET_INPUT(percept)
rule <-- RULE_MATCH(state, rules)
action <-- RULE_ACTION[rule]
return action
```

- Simple-reflex agents are simple, but they turn out to be of very limited intelligence
- The agent will work only if the correct decision can be made on the basis of the current percept – that is only if the environment is fully observable
- Infinite loops are often unavoidable – escape could be possible by randomizing

Utility-based agents



Utility-based agents

- Goals alone are not really enough to generate high quality behavior in most environments – they just provide a binary distinction between happy and unhappy states
- A more general performance measure should allow a comparison of different world states according to exactly how happy they would make the agent if they could be achieved
- Happy – Utility (the quality of being useful)
- A utility function maps a state onto a real number which describes the associated degree of happiness