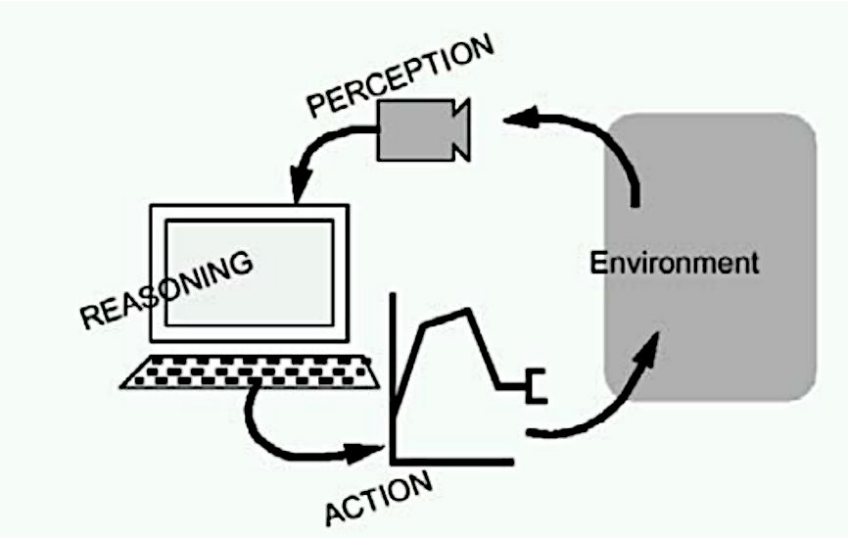
Agents and Environments

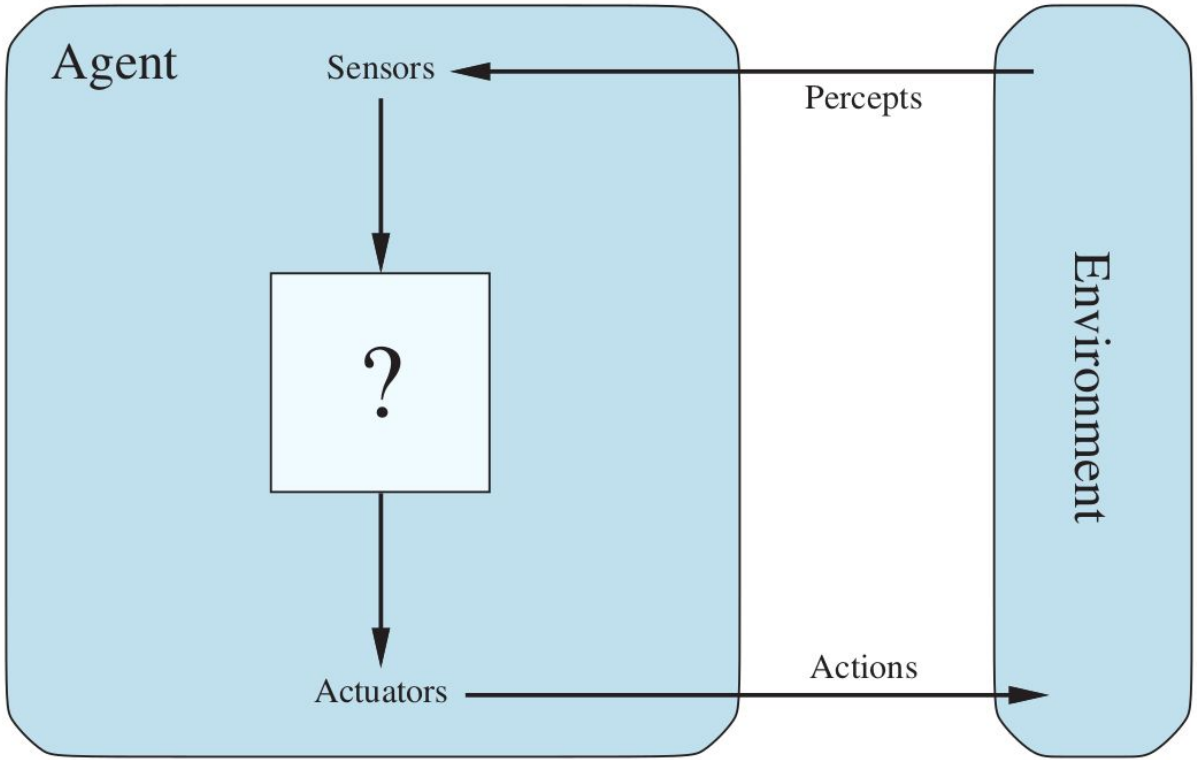
Overview of an AI System

- An agent is anything that can be viewed as “perceiving” its environment through “sensors” and acting upon that environment through “actuators”

Perception-Action Cycle



Artificial Intelligence (Agent View)



Agents interact with environments through sensors and actuators.

Artificial Intelligence (Agent View)

- An agent is anything that can be viewed as "perceiving" its environment through "sensors" and acting upon that environment through "actuators"
- **Human agent:**
  - Sensors:
  - Actuators:

Artificial Intelligence (Agent View)

- An agent is anything that can be viewed as "perceiving" its environment through "sensors" and acting upon that environment through "actuators"
- **Human agent:**
  - Sensors: eyes, ears, and other organs
  - Actuators: hands, legs, mouth, and other body parts

Artificial Intelligence (Agent View)

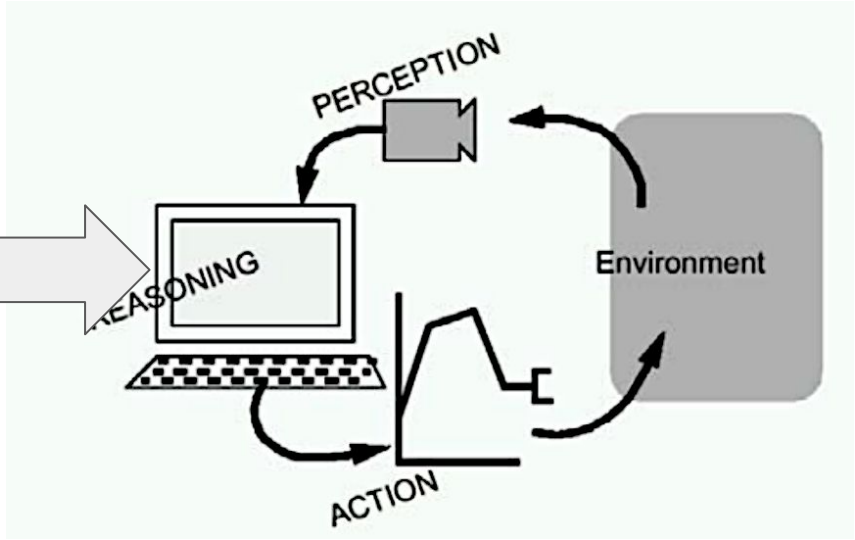
- An agent is anything that can be viewed as "perceiving" its environment through "sensors" and acting upon that environment through "actuators"
- Human agent:
  - Sensors: eyes, ears, and other organs
  - Actuators: hands, legs, mouth, and other body parts
- **Robotic agent:**
  - Sensors:
  - Actuators:

Artificial Intelligence (Agent View)

- An agent is anything that can be viewed as "perceiving" its environment through "sensors" and acting upon that environment through "actuators"
- Human agent:
  - Sensors: eyes, ears, and other organs
  - Actuators: hands, legs, mouth, and other body parts
- **Robotic agent:**
  - Sensors: cameras and laser range finders
  - Actuators: various motors, mechanical arms

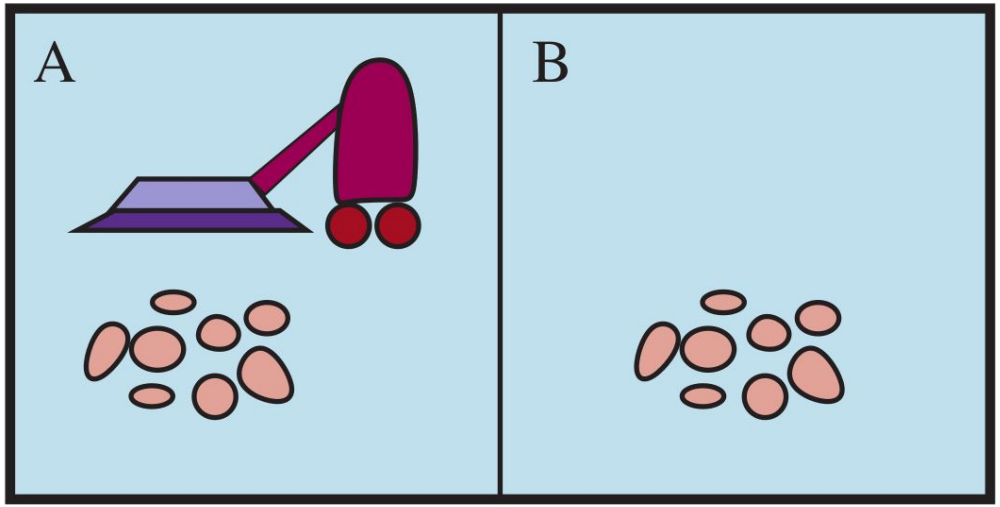
Perception-Action Cycle

Natural Language Processing
Reasoning: Knowledge Representation
Decision-Making (search, planning, decision theory)
Reasoning Processes (logical, probabilistic)
Machine Learning, Neural Networks



Vacuum- cleaner example

- Actuators: wheels, different brushes, vacuum extractor.
- Sensors: camera, dirt detection sensor, etc.
- Percepts: location and contents e.g., [A, Dirty]
- Actions: Left, Right, Suck, NoOp
- Agent function: mapping from percepts to actions.
- Environment: room.
- Performance: cleanness, battery life



Vacuum- cleaner example

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*
⋮	⋮

Rationality

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

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

Rational Agent

Given the ***percept sequence*** and whatever **built-in knowledge** the agent has, a *rational agent* should select an ***action*** that maximize the expected value of its ***performance* measure**,.

Rational Agent

Vacuum-cleaner agent: clean a square if it is dirty and moves to the other square if not.

Is vacuum-cleaner agent a rational agent?

Rational Agent

Vacuum-cleaner agent: clean a square if it is dirty and moves to the other square if not.

Is vacuum-cleaner agent a rational agent?

Yes it is a rational agent.

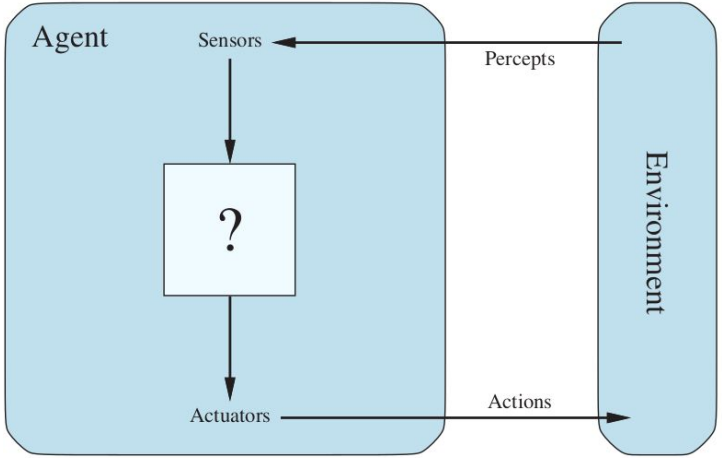
Structure of Agents

- **Agent program:** that implements the agent function the mapping from percepts to actions.
- **Agent architecture**: We assume Agent program will run on some sort of computing device with physical sensors and actuators

Agent = Architecture + Program

Agent function and program

- An agent is completely specified by the agent function mapping percept sequences to actions
- (In principle, one can supply each possible sequence to see what it does. Obviously, a lookup table would usually be immense.)
- One agent function (or a small class) is rational
- **Aim**: find a way to implement the rational agent function concisely



Agent programs

function REFLEX-VACUUM-AGENT([*location*,*status*]) **returns** an action

if
else if
else if

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*
⋮	⋮

Agent programs

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*

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*
⋮	⋮

PAGE description.

(Percept, Actions, Goals, and Environment)

PAGE

In designing intelligent systems there are four main factors to consider:

- P Percepts – the inputs to our system
- A Actions – the outputs of our system
- G Goals – what the agent is expected to achieve
- E Environment – what the agent is interacting with

PAGE

In designing intelligent systems there are four main factors to consider:

- P Percepts –
- A Actions –
- G Goals –
- E Environment –

PAGE

In designing intelligent systems there are four main factors to consider:

- P Percepts – video, accelerometers, engine sensors, keyboard, GPS, ...
- A Actions – steer, accelerate, brake, horn, speak/display, ...
- G Goals – Safety, time, legal drive, comfort., ...
- E Environment – Roads, other cars, pedestrians, road signs.

Task Environment specification

(Performance, Environment, Actuators, Sensors).

What is PEAS for a self-driving car?

- **Performance**:
- **Environment**:
- **Actuators**:
- **Sensors**:

What is PEAS for a self-driving car?

- **Performance**: Safety, time, legal drive, comfort.
- **Environment**: Roads, other cars, pedestrians, road signs.
- **Actuators**: Steering, accelerator, brake, signal, horn.
- **Sensors**: Camera, sonar, GPS, Speedometer, odometer, accelerometer, engine sensors, keyboard.

Environment (types and examples)

Environment Types

- **Fully observable and partially observable**:
  - An agent's sensors give it access to the complete state of the environment at each point in time (e.g., tic tac toe).
  - An environment might be partially observable because of noisy and inaccurate sensors (e.g., self driving car environment).
  - If the agent has no sensors at all then the environment is unobservable.

Environment Types

- **Deterministic and stochastic**:
  - Deterministic: next state of the environment is completely determined by current state and action executed by agent,e.g., chess (otherwise nondeterministic)
  - Stochastic: If it explicitly deals with probabilities (e.g., there's a 25% chance of rain tomorrow, and snake and ladder).
  - Nondeterministic: If the possibilities are listed without being quantified (e.g., there's a chance of rain tomorrow).

Environment Types

- **Episodic and sequential**:
  - **Episodic**: agent's experience is divided into atomic "episodes" and choice of action in each episode depends only on the episode itself. Episode consists of agent perceiving and then performing a single action (e.g., classification)
  - **Sequential:** current decision could affect all future decisions (e.g., Chess and taxi driving are sequential).

Environment Types

- **Static and dynamic**:
  - **Dynamic:** state of world can spontaneously change without any action by agent. e.g., agent can plan an optimal driving route from A to B, but an accident or unusually bad rush hour traffic can spoil the plan.
  - **Static**: The environment is unchanged while an agent is deliberating. Static environments are easy to deal with. Crossword puzzles are static.
  - **Semidynamic**: environment itself does not change with time but the agent's performance score does, then we say the environment is semidynamic. Chess, when played with a clock, is semidynamic.

Environment Types

- **Discrete and continuous**:
  - discrete/continuous distinction applies to the state of the environment, to the way time is handled, and to the percepts and actions of the agent.
  - Chess environment has a finite number of distinct states (discrete)
  - Taxi driving is a continuous-state and continuous-time.

Environment Types

- **Single agent and multiagent**:
  - Single agent: agent operating by itself in an environment, e.g., an agent solving a crossword puzzle.
  - Multiagent: multiple agents operating in an environment. E.g., chess is a competitive multiagent environment

Environment Types

Known vs. unknown:

- the distinction refers not to the environment itself but to the agent's state of knowledge about the *"laws of physics"* of the environment.
- **Known environment**: the outcomes (or outcome probabilities if the environment is nondeterministic) for all actions are given.
- **Unknown**: the agent will have to learn how it works in order to make good decisions.

Task environments examples

Environment	Observable	Agents			
8-puzzle	Fully	Single	Deterministic	Static	Discrete
Chess	Fully	Multi	Deterministic	(Semi)Static	Discrete
Pocker	Partially	Multi	Stochastic	Static	Discrete
Car	Partially	Multi	Stochastic	Dynamic	Continuous

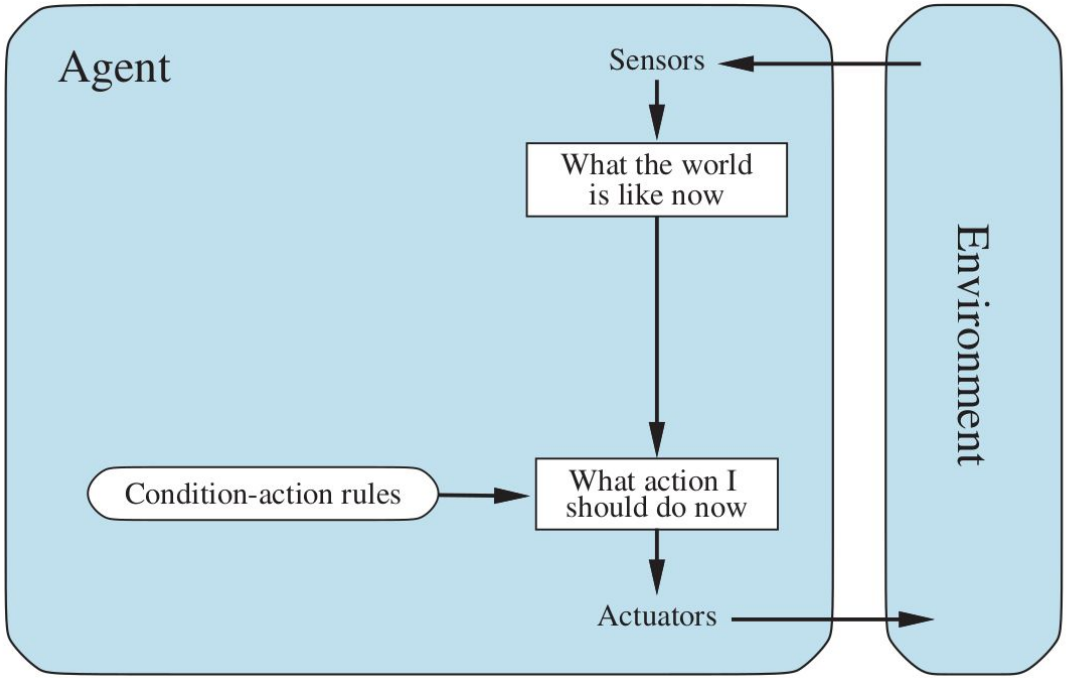
Agents Types

Agents Types

1. Simple Reflex Agents
2. Reflex Agents with an Internal State
3. Goal based agents
4. Utility based agents

Simple Reflex Agents

Simple Reflex Agents work if the environment is fully observable



Simple Reflex Agents

- Simple reflex agents act only on the basis of the current percept, ignoring the rest of the percept history.
- Finding the Action appropriate for a given set of Percepts in a look-up table is clearly going to be impossible because of the high number of entries such a table would require.
- Summarise portions of the look-up table by noting commonly occurring input/output associations which can be written as **condition-action rules**

if *{set of percepts}* **then** *{set of actions}*

Simple Reflex Agents

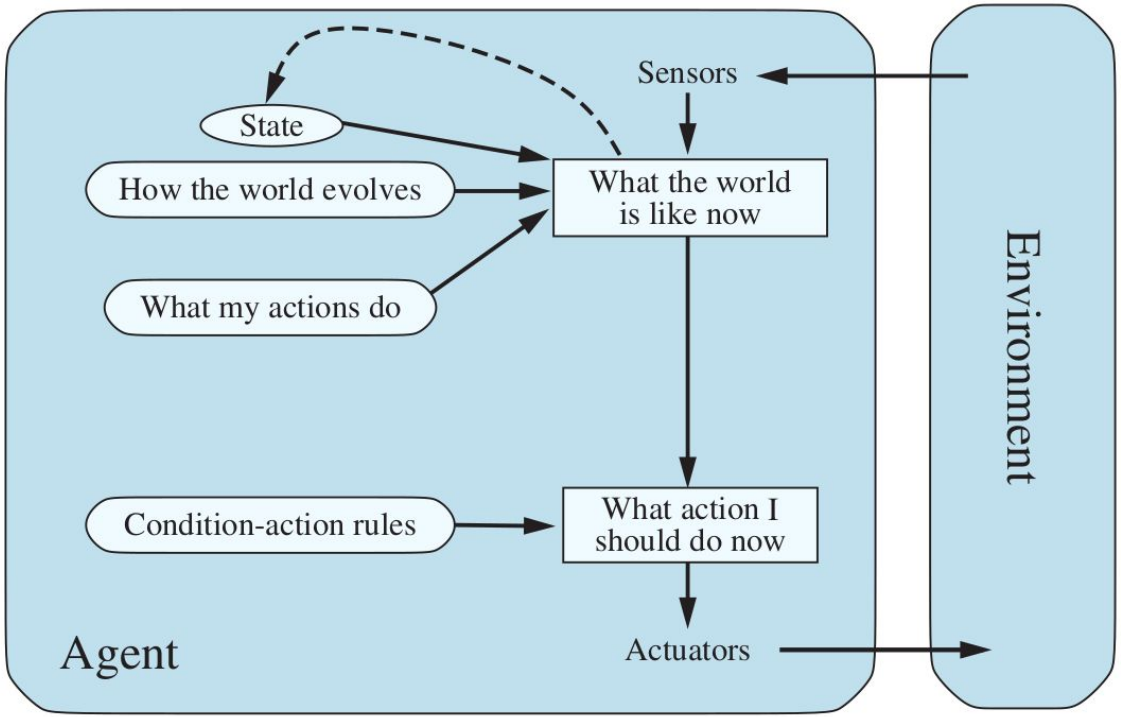
- Simple reflex agents act only on the basis of the current percept, ignoring the rest of the percept history.
- Finding the Action appropriate for a given set of Percepts in a look-up table is clearly going to be impossible because of the high number of entries such a table would require.
- Summarise portions of the look-up table by noting commonly occurring input/output associations which can be written as **condition-action rules**

Condition–action rule example:

if car-in-front-is-braking ***then*** initiate-braking.

Reflex Agents with an Internal State

Handle partial observability by keeping track of the part of the world it can't see now.

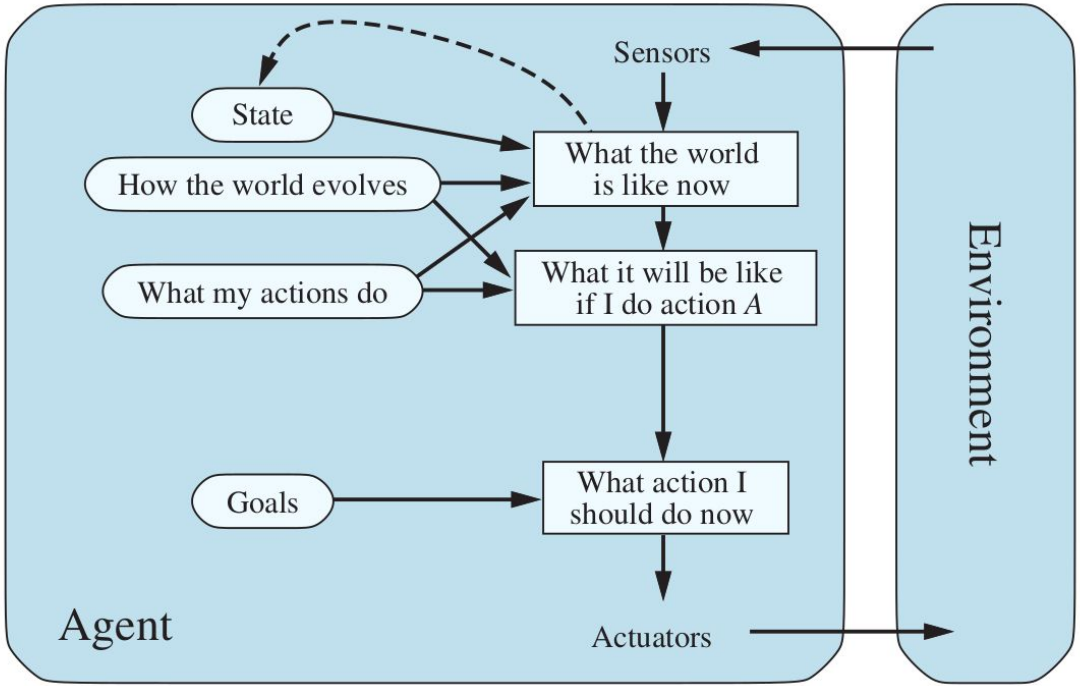


Reflex Agents with an Internal State

- One problem with Simple Reflex Agents is that their actions depend only on the current information provided by their sensors.
- A model-based reflex agent should maintain some sort of **internal model** that depends on the percept history and thereby reflects at least some of the unobserved aspects of the current state.
- The knowledge about "how the world works" is called a **model of the world**.
- A model-based agent can handle **partially observable** environments.

Goal Based Agents

Knowing the current state of the environment is not enough. The agent needs some goal information.

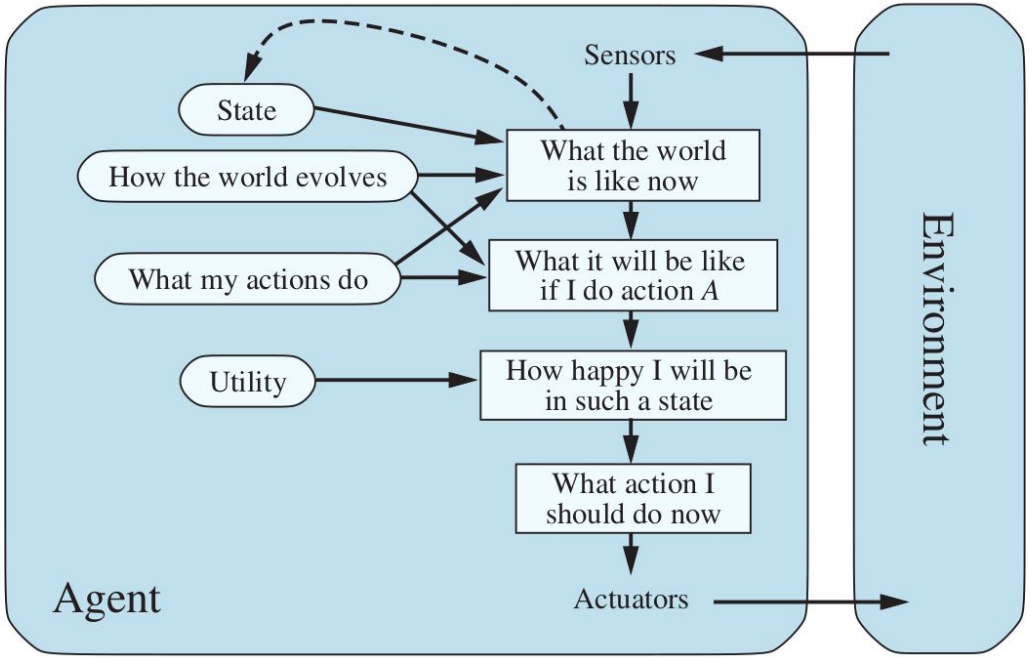


Goal Based Agents

- Goal-based agents further expand on the capabilities of the model-based agents, by using "goal" information. Goal information describes situations that are desirable.
- The appropriate action for the agent will often depend on what its goals are, and so it must be provided with some goal information.
- If a long sequence of actions is required to reach the goal, then Search and Planning are the sub-fields of AI that must be called into action.
- In simple reflex agents, such information would have to be pre-computed and built into the system by the designer.

Utility Based Agents

Sometimes achieving the desired goal is not enough. We may look for quicker, safer, cheaper trip to reach a destination.

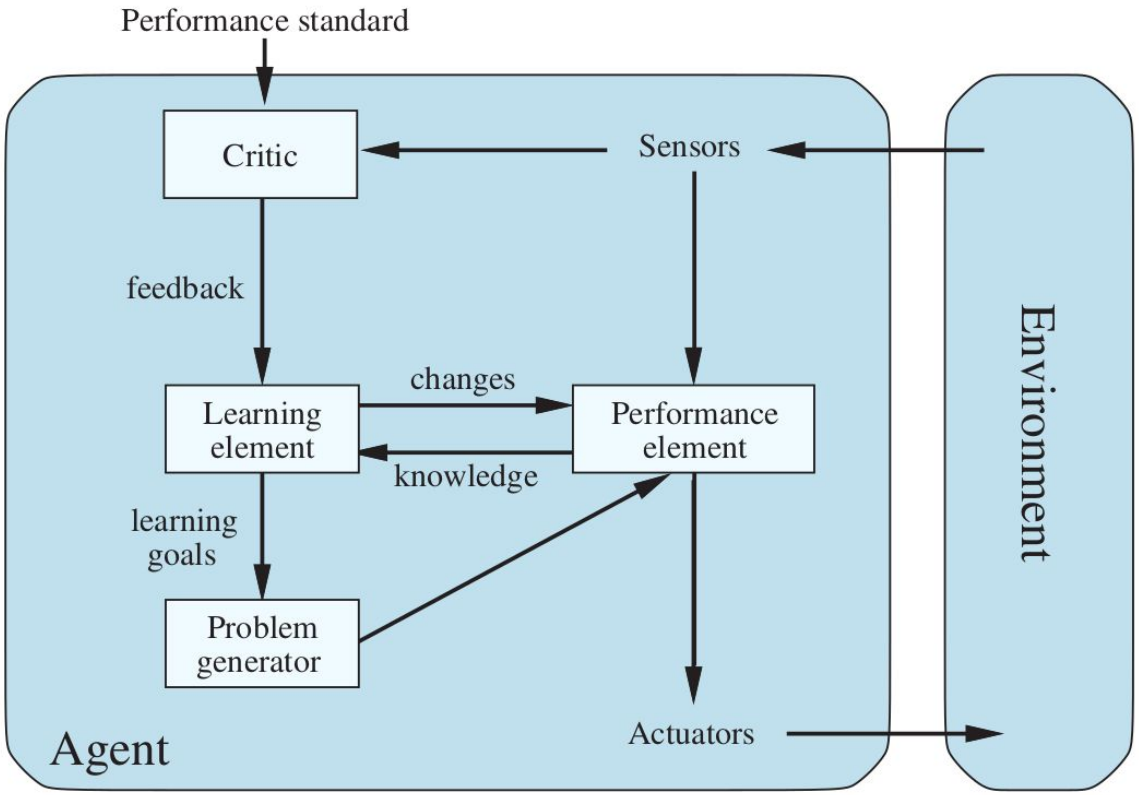


Utility Based Agents

- Utility function maps a state to a measure of the utility (or usefulness) of the state. It helps to choose between alternative sequences of actions/states that lead to a given goal being achieved.
- When there are conflicting goals, only some of which can be achieved, the utility is able to quantify the appropriate trade-offs.
- When there are several goals, none of which can be achieved with certainty, the utility allows the likeliness of success to be weighed up against the importance of the goals.
- A rational utility-based agent chooses the action that maximizes the expected utility of the action outcomes.

General learning agent

Learning has the advantage that it allows the agents to initially operate in unknown environments and to become more competent than its initial knowledge alone might allow.



Learning Agents

A general learning agent has four basic components:

- The Performance Element – which takes in percepts and decides on appropriate actions in the same way as a non-learning agent.
- The Critic – which uses a fixed standard of performance to tell the learning element how well the agent is doing.
- The Learning Element – that receives information from the critic and makes appropriate improvements to the performance element.
- The Problem Generator – that suggests actions that will lead to new and informative experiences (e.g. as in carrying out experiments).

Summary of Agents

- Simple reflex agents respond directly to percepts
- Model-based reflex agents maintain internal state to track aspects of the world that are not evident in the current percept.
- Goal-based agents act to achieve their goals
- Utility-based agents try to maximize their own expected "happiness."
- Learning Agents are able to learn by actively exploring and experimenting with their environment.
- All Agents can improve their performance through learning.