Advanced Artificial Intelligence

Intelligent Agents Discusion

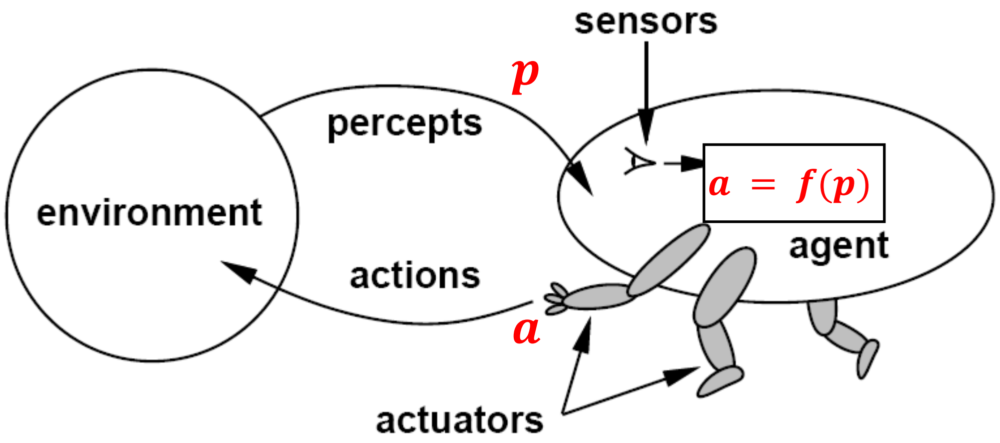
Image: "Robot at the British Library Science Fiction Exhibition" by BadgerGravling

Module Review 1

Agent Function and Agent Program

The agent function maps from the set of all possible *percept sequences* P^* to the *set of actions* A formulated as an abstract mathematical function.

$$f : P^* \rightarrow A$$



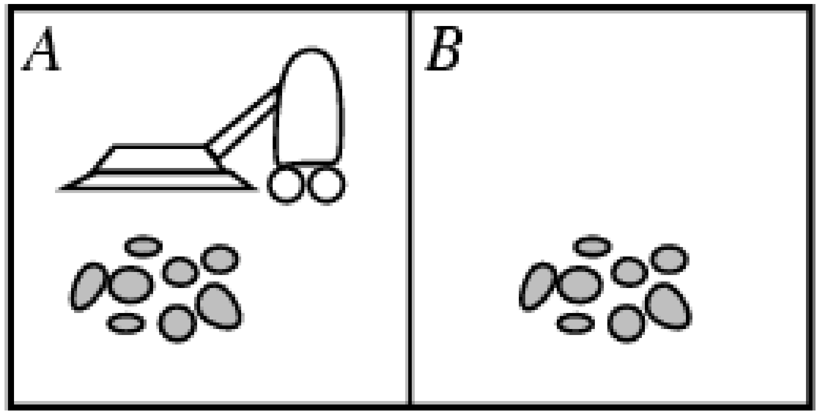
The agent program is a concrete implementation of this function for a given physical system.

Agent = architecture (hardware) + agent program (implementation of f)

- Sensors
- Memory
- Computational power

Example: Vacuum-cleaner World

- **Percepts:**
  Location and status,
  e.g., [A, Dirty]
- **Actions:**
  Left, Right, Suck, NoOp



Most recent Percept p

Agent function: $f : P^* \rightarrow A$

Percept Sequence	Action
[A, Clean]	Right
[A, Dirty]	Suck
…	
[A, Clean], [B, Clean]	Left
…	
[A, Clean], [B, Clean], [A, Dirty]	Suck
…	

Implemented agent program:

function Vacuum-Agent([location, status])
returns an action a

```
if status = Dirty then return Suck
else if location = A then
                          return Right
else if location = B then
                          return Left
```

Problem: This table can become infinitively large!

Rational Agents

Rule: Pick the action that maximize the expected utility

$$a = \operatorname{argmax}_{a \in \mathrm{A}} E(U \mid a)$$

This means:

- **Rationality is an ideal** – it implies that no one can build a better agent
- **Rationality ≠ Omniscience** – rational agents can make mistakes if percepts and knowledge do not suffice to make a good decision
- **Rationality ≠ Perfection** – rational agents maximize **expected** outcomes not actual outcomes
- **It is rational to explore and learn** – I.e., **use percepts** to supplement prior knowledge and become autonomous
- **Rationality is often bounded** by available memory, computational power, available sensors, etc.

Some Environment Types Revisited

Fully observable: The agent's sensors always show the whole **state**.

vs.

Partially observable: The agent only perceives part of the **state** and needs to remember or infer the rest.

Deterministic:

a) **Percepts** are 100% reliable.

b) Changes in the environment are completely determined by the current **state** of the environment and the agent's **action**.

vs.

Stochastic:

a) **Percepts** are unreliable (noise distribution, sensor failure probability, etc.). This is called a stochastic sensor model.

b) The **transition function** is stochastic leading to transition probabilities and a Markov process.

Known: The agent knows the **transition function**.

vs.

Unknown: The needs to **learn the transition function** by trying actions.

We will spend the whole course on discussing algorithms that can deal with environments that have different combinations of these three properties.

Case Study: Self-Driving Cars

Self-driving Cars

SAE Automation Levels

- Level 1 - Driver Assistance (“hands on”)
- Level 2 - Partial Automation (“hands off”)
- Level 3 - Conditional Automation
- Level 4 - High Automation
- Level 5 - Full Automation (“steering wheel optional”)

Components

- Sensing
- Maps
- Path planning
- Controlling the vehicle

Why is this so hard?

A Self-Driving Car as a Rational Agents

Rule: Pick the action that maximize the expected utility

$$a = \text{argmax}_{a \in \mathrm{A}} \, E(U \mid a)$$

Answer the following questions:

- If we have two cars and one provides more (expected) utility. Which car is rational?
- Can a rational self-driving car be involved in an accident?
- How would a self-driving car explore and learn?
- What does bounded rationality mean for a self-driving car?

PEAS Description of the Environment of a Self-Driving Car

Complete the PEAS description.

Performance measure	Environment	Actuators	Sensors

Environment for a Self-Driving Car

☐ **Fully observable:** The agent's sensors always show the whole **state**.

vs.

☐ **Partially observable:** The agent only perceives part of the **state** and needs to remember or infer the test.

Deterministic:

☐ **a)** **Percepts** are 100% reliable

☐ b) Changes in the environment are completely determined by the current **state** of the environment and the agent's **action**.

vs.

Stochastic:

☐ **a)** **Percepts** are unreliable (noise distribution, sensor failure probability, etc.). This is called a stochastic sensor model.

☐ b) The **transition function** is stochastic leading to transition probabilities and a Markov process.

☐ **Known:** The agent knows the **transition function**.

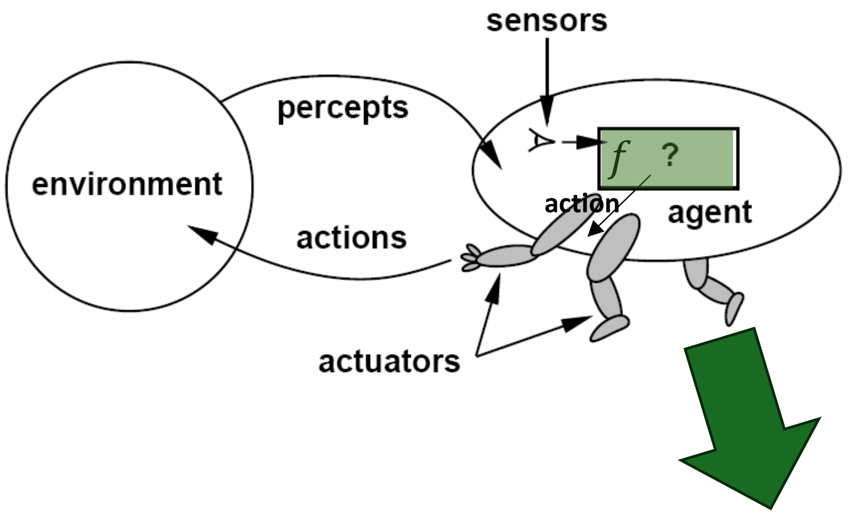
vs.

☐ **Unknown:** The needs to **learn the transition function** by trying actions.

☑ Check what applies and explain what it means for a self-driving car.

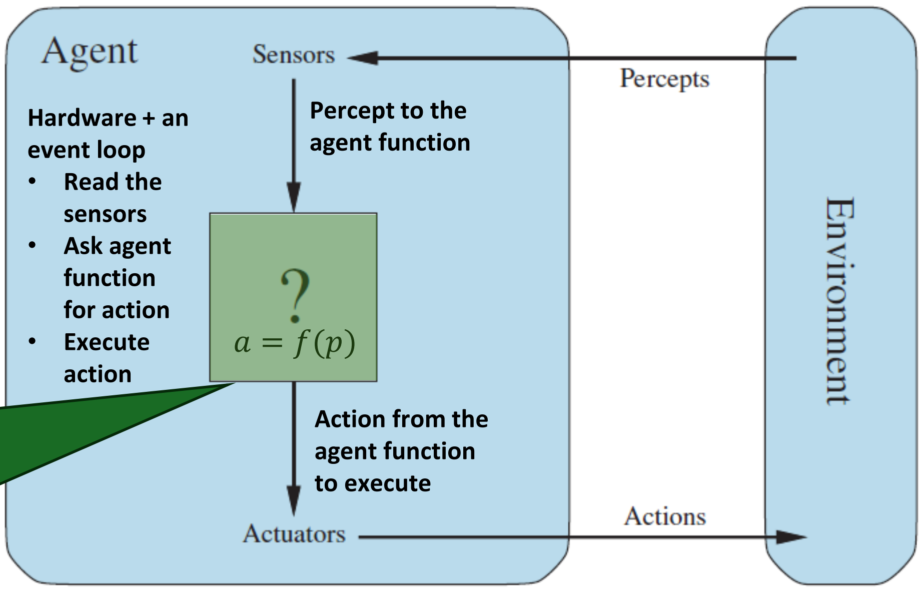
Module Review 2

Designing a Rational Agent



Remember the definition of a rational agent:

> *"For each possible percept sequence, a rational agent should select an* ***action*** *that* ***maximizes its expected performance measure****, given the evidence provided by the* ***percept sequence*** *and the* ***agent's built-in knowledge****."*



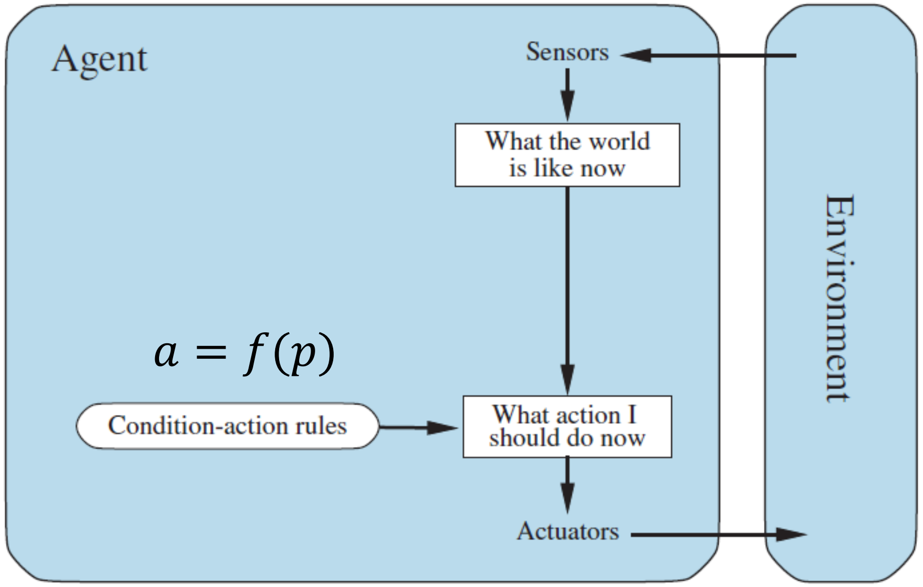
Agent Function

- Represents the "brain"
- Assess performance measure
- Remember percept sequence
- Built-in knowledge

Important: Everything outside the agent function represents the environment. This includes the physical robot, its sensors and its actuators, and event loop!

Simple Reflex Agent

- Uses only built-in knowledge in the form of **rules** that select action only **based on the current percept.** This is typically very fast!
- The **agent does not know about the performance measure**! But well-designed rules can lead to good performance.
- The agent needs no memory and ignores all past percepts.

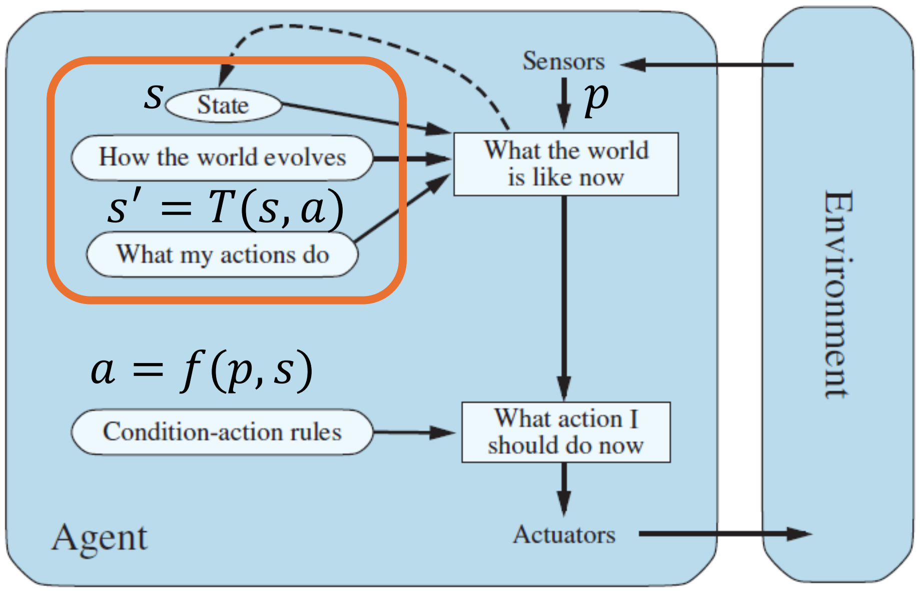


The interaction is a sequence: $p_0, a_0, p_1, a_1, p_2, a_2, \ldots p_t, a_t, \ldots$

Example: A simple vacuum cleaner that uses rules based on its current sensor input.

Model-based Reflex Agent

- Maintains a **state variable** to keeps track of aspects of the environment that cannot be currently observed. I.e., it has memory.
- It knows how the environment evolves over time given its last action. It updates the state using a **transition function** and the new percept.
- There is now more information for the **rules** to make better decisions.



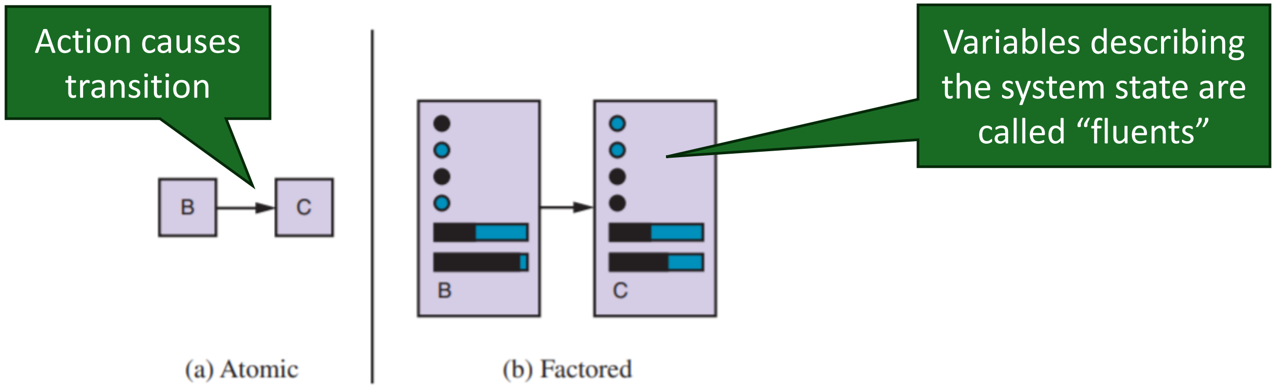
The interaction is a sequence: $p_0, s_0, a_0, p_1, s_1, a_1, p_2, s_2, a_2, p_3, \dots, p_t, s_t, a_t, \dots$

Example: A vacuum cleaner that remembers were it has already cleaned.

State Representation

States help to keep track of the environment and the agent in the environment. This is often also called the **system state**. The representation can be

- **Atomic**: Just a label for a black box. E.g., A, B
- **Factored**: A set of attribute values called fluents.
  E.g., [location = left, status = clean, temperature = 75 deg. F]

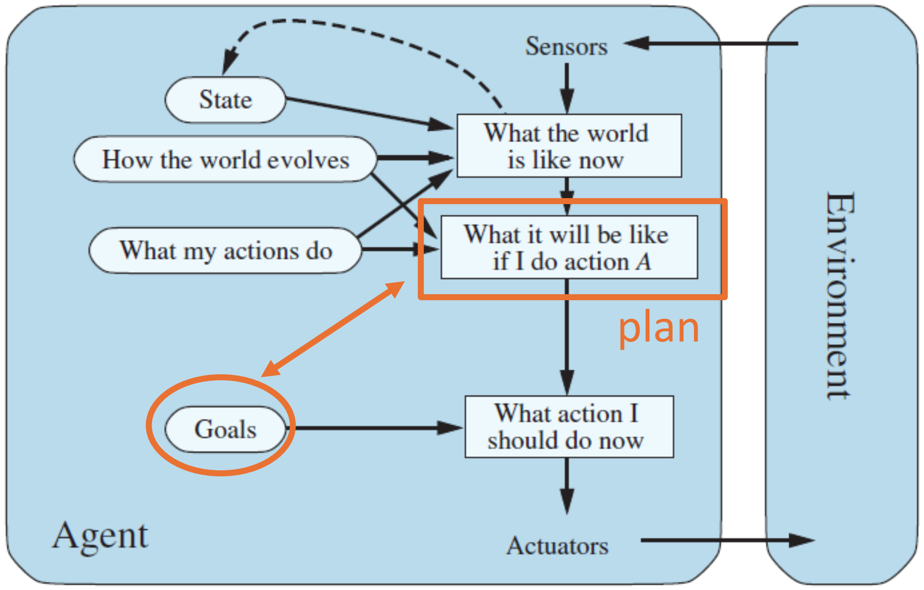


We often construct atomic labels from factored information. E.g.: If the agent's state is the coordinate x = 7 and y = 3, then the atomic state label could be the string "(7, 3)". With the atomic representation, we can only compare if two labels are the same. With the factored state representation, we can reason more and calculate the distance between states!

The set of all possible states is called the **state space S**. This set is typically very large!

Goal-based Agent

- The agent has the task of reaching a defined **goal state** and is then finished.
- The agent needs to move towards the goal. As special type is a **planning agent** that uses **search algorithms** to plan a sequence of actions that leads to the goal.
- Performance measure: the **cost to reach the goal**.


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
plan
Goals
What action I should do now
Agent
Actuators
Environment

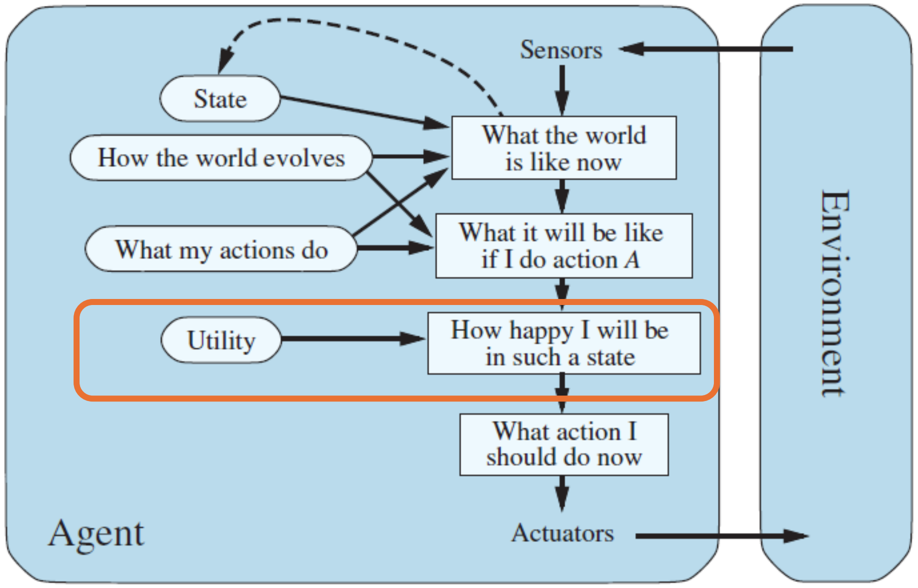
$$a = \operatorname{argmin}_{a_0 \in A} \underbrace{\left[\sum_{t=0}^{T} c_t \,\middle|\, s_T \in S^{goal} \right]}_{\text{Sum of the cost of a planed sequence of actions that leads to a goal state}}$$

The interaction is a sequence: $\underbrace{p_0, s_0, a_0, p_1, s_1, a_1, p_2, s_2, a_2, \dots, s^{goal}}_{\text{cost}}$

Example: Solving a puzzle. What action gets me closer to the solution?

Utility-based Agent

- The agent uses a utility function to evaluate the **desirability of each possible states.** This is typically expressed as the reward of being in a state $R(s)$.
- Choose actions to stay in desirable states.
- Performance measure: The discounted sum of **expected utility over time**.


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
Agent
Actuators
Environment

$$a = \operatorname{argmax}_{a_0 \in \mathrm{A}} E \underbrace{\left[\sum_{t=0}^{\infty} \gamma^t r_t \,\middle|\, a_0 \right]}_{\text{Implements rational behavior: Utility is the expected future discounted reward}}$$

Techniques: Markov decision processes, reinforcement learning

The interaction is a sequence: $\underbrace{p_0, s_0, a_0, p_1, s_1, a_1. p_2, s_2, a_2, \ldots}_{\text{reward}}$

Example: An autonomous Mars rover prefers states where its battery is not critically low.

Case Study: Self-Driving Cars

State Representation: Self-Driving Car

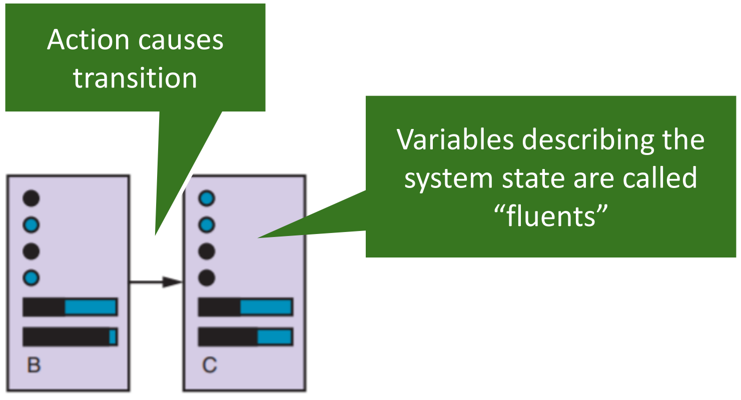
States help to keep track of the environment and the agent in the environment.

Design a structured representation for the state of a self-driving car.

a) What fluents should it contain?

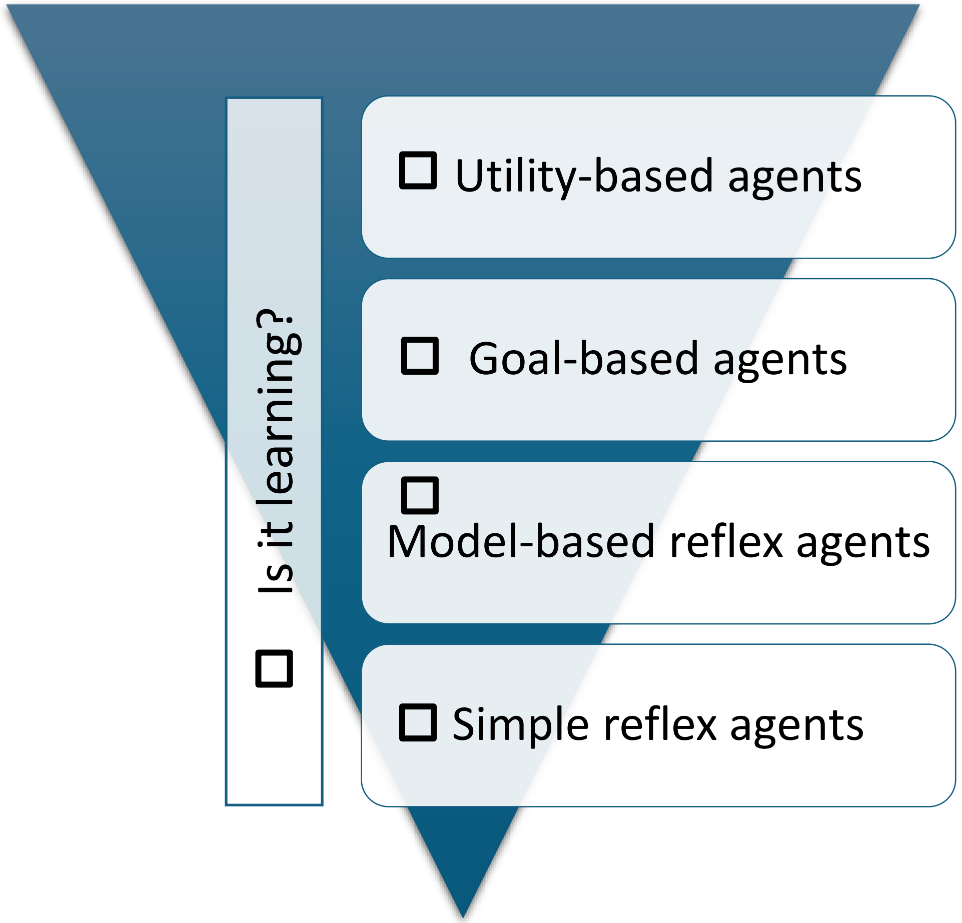
b) What actions can cause transitions?

c) Draw a small transition diagram.



(b) Factored

What Type of Intelligent Agent is a Self-Driving Car?



Does it collect utility over time? How would the utility for each state be defined?

Does it have a goal state?

Does it store state information. How would they be defined (atomic/factored)?

Does it use simple rules based on the current percepts?

☑ Check what applies

Why is this so hard?

- Self-driving cars operate in a very complicated partially observable, stochastic, and dynamic environment.
- Can only use bounded rationality because of limits with sensors and computational power.
- Require a set of different agents that cooperate.