Introduction to Artificial Intelligence

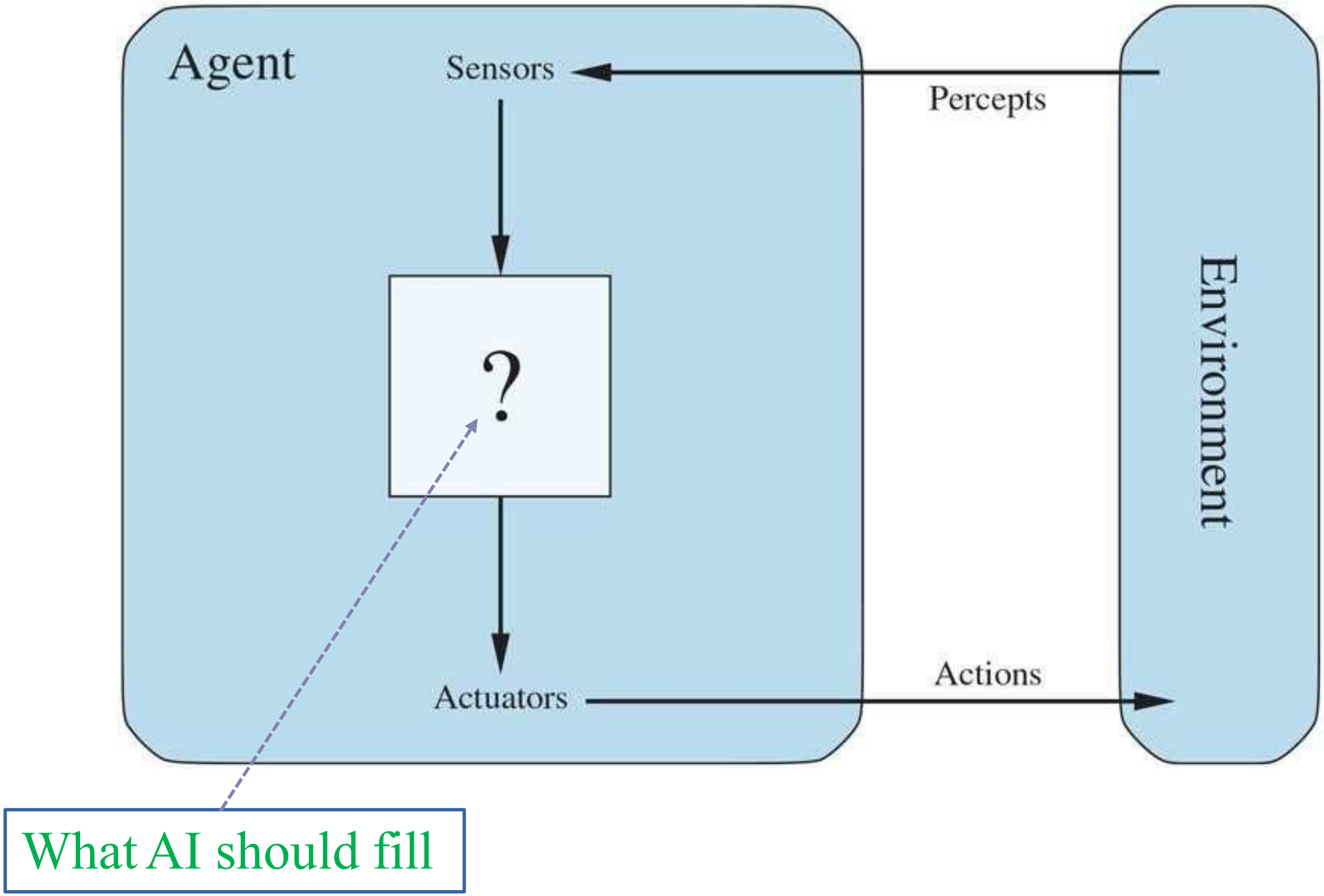
Subject 3: Intelligent Agents

Dr. Mohammed Shambour

Intelligent Agents

- What is an agent?
  - **An agent** is any entity that **perceives** its environment through sensors and acts upon that environment through actuators.
  - **Examples:**
    - **A human agent uses eyes and ears as sensors, and hands and voice as actuators.**
    - **A robot agent uses cameras and infrared sensors, and acts through motors and grippers.**
    - **A thermostat senses temperature and acts by switching the heating/cooling system on or off.**
  - The **environment** could be everything—the part of the world that is relevant to the agent.

Diagram of an agent



Percept & Percept sequence

- Percept
  - An agent's immediate perceptual input at any specific moment—whatever its sensors are detecting instantly.
- Percept sequence
  - Complete history of everything that the agent has ever perceived.

"*an agent's choice of action at any given instant can depend on its built-in knowledge and on the entire percept sequence observed to date, but not on anything it hasn't perceived.*"

Agent function & program

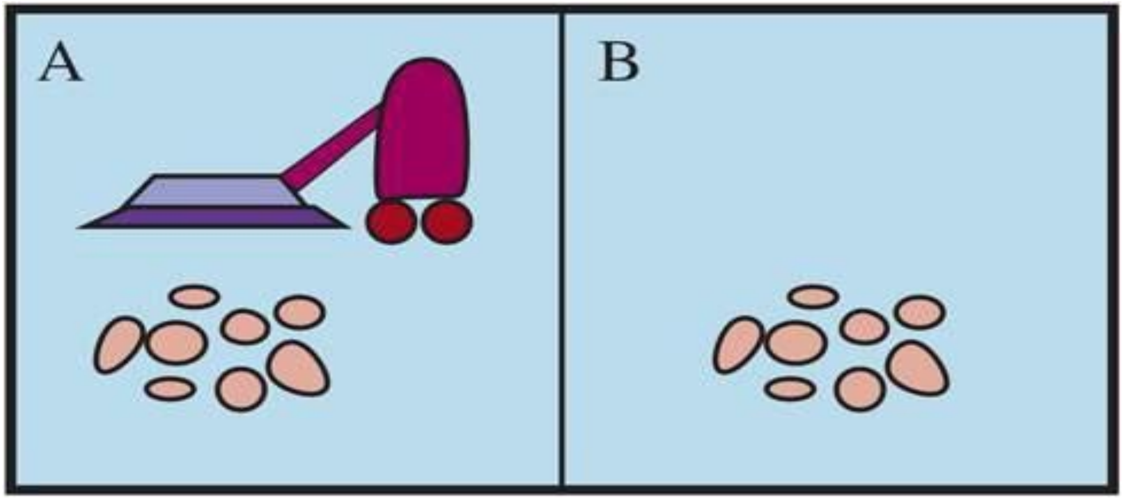
- **Agent Function:** The **mathematical description** of an agent's behavior. It is defined as a function that **maps** the complete history of everything the agent has ever perceived (the percept sequence) to a single, immediate action.

- **Agent Program:** The **practical implementation** of this function - the actual code or algorithm running on physical hardware that realizes the agent's behavior.

- **Example:**
  - **Agent Function:** "If temperature > 30°C, turn on AC; else turn off AC".
  - **Agent Program:** The actual software running on the thermostat's microcontroller that reads temperature sensors and controls the AC unit.

Example: Vacuum-cleaner world

- **Perception**: Where it is in? Clean or Dirty?
- **Actions**: Move left, Move right, suck up dirt, do nothing



Environment Description

•There are **two locations**:

- Square **A**
- Square **B**

•Each location can be either:

- **Clean**, or
- **Dirty**

•The **vacuum agent** can:

- **Move Left** or **Move Right**
- **Suck up dirt** (clean the current square)

Example: Vacuum-cleaner world

- 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 below

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*
⋮	⋮

How It Works Step-by-Step

1. The agent perceives **its location** and **the status (Clean/Dirty)**.
2. If it's **Dirty**, it performs **Suck**.
3. If it's **Clean**, it **moves** to the other location.
4. It repeats this process indefinitely until both squares are clean.

The Algorithm's Implementation

Function Reflex-Vacuum-Agent([*location*,*status*])   return an action

If $status =\ Dirty$ **then return** *Suck*

else if $location = A$ **then return** *Right*

else if $location = B$ **then return** *left*

Good Behavior: The Concept of Rationality

- Rational agent
  - **A rational agent** is one that chooses the action that is expected to maximize its Performance Measure, based on the complete sequence of percepts it has observed so far (its history).

- What makes an action "correct" or "rational"?
  - An action is correct if it is the one that **causes the agent to be most successful** according to some **predefined performance measure.**
  - So, we need ways to measure success.

Consequentialism in AI

- AI has generally adhered to a concept called **consequentialism**: ***we evaluate an agent's behavior by its consequences***.
- When an agent is placed 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.**

We don't care ***why*** the agent chose an action; we only care about the ***result*** of that action in the real world.

Performance measure

- **Performance measure:**

  The performance metric (or objective function) is the crucial formal definition of success for an AI agent.

- An agent, based on its percepts → action sequence:

  if desirable, it is said to be performing well.

- There is **No** universal performance measure for all agents.
- Humans have desires and preferences of their own, but Machines do not have desires and preferences of their own, they only have the objective function.

Performance measure

- **The General Rule for Designing Performance Measures**
  The fundamental rule is to design the performance measure based on the **end result** (the sequence of desirable environment states), not the agent's internal workings or behavior pattern.

- **Example: The Vacuum Cleaner World:**
  - **Goal**: We want the floor clean (the desired end state).
  - **Incorrect Measure**: A measure based on, say, "number of times the agent turns left“.
  - **Correct Measure:** A measure based on the percentage of the floor that is dirt-free and perhaps minimizing time or noise.
  - **Implication:** We should evaluate an agent based on its **goal** achievement, not its **methods**. If an agent discovers an unconventional but highly effective approach that successfully accomplishes the objective—such as finding an unexpected way to clean floors more efficiently—this represents rational and successful behavior.

The Omniscient Agent vs rational agent

- **The Omniscient Agent**

  An omniscient agent is a hypothetical, perfect agent that:

  - ❑ Knows the actual outcome of its actions in advance.
  - ❑ Assumes no other possible outcomes exist (it has perfect foresight).

  In short, it knows the future with 100% certainty. This is impossible in reality because all real-world environments involve uncertainty, complexity, and unpredictability.

- **A rational agent** operates in the real world, which is by nature uncertain. Because it cannot know the future, a rational agent must maximize its expected performance, not its actual performance.

Learning

- Does a rational agent depend on only current percept?
  - No, the past percept sequence should also be used
  - This is called **learning**
  - After performing a task, the agent should modify its behavior to achieve better performance when faced with the same task in the future.

Autonomy

- If an agent just relies on the prior knowledge of its designer rather than its own percepts then the agent lacks ***autonomy.***

A rational agent should be ***autonomous- it should learn what it can to compensate for partial or incorrect prior knowledge.***

- E.g., a clock
  - No input (percepts)
  - Run only but its own algorithm (prior knowledge)
  - No learning, no experience, etc.
- E.g., a vacuum cleaner that learns to predict where and when extra dirt will appear will perform better than a cleaner that doesn't..
- It would be reasonable to provide an AI agent with some initial knowledge as well as an ability to learn. After sufficient experience of its environment, the behavior of a rational agent can become effectively **independent** of its prior knowledge.

Software Agents (Softbots)

Software agents, often called **softbots**, are AI programs that exist and operate within a purely **virtual or software environment**. They are the opposite of physical agents (like robots) that operate in the real world.

Key Characteristics:

1. **Virtual Environment:** Softbots work entirely within environments like the internet, operating systems, databases, or virtual games.
2. **Percepts and Actions:**
   - **Percepts** are typically digital signals, such as file contents, network packets, function calls, or data retrieved from a database.
   - **Actions** are digital operations, such as deleting a file, sending an email, changing a database entry, or displaying information.
3. **Examples:**
   - **Web Crawlers/Spiders:** Agents that autonomously browse and index web pages.
   - **Chatbots and Virtual Assistants:** Agents that interact with users using natural language.
   - **Search Engines:** Agents that retrieve and rank information based on user queries.

Task environments

- <u>Task environments are the problems</u> (The task environment defines what needs to be done — the problem the agent must solve.)
- <u>While the rational agents are the solutions</u> (The rational agent defines how it acts — the solution that perceives, decides, and acts to achieve its goals in that environment.)
- Specifying the task environment
  - PEAS description as fully as possible
    - Performance
    - Environment
    - Actuators
    - Sensors
- Before designing an intelligent agent, always **specify the task environment fully** using the PEAS framework — because understanding the problem is the first step to building a rational solution.

Automated taxi driver as an example

PEAS Component	Automated Taxi Driver Example
Performance	Safety (low collision rate), legality (low fines), comfort (smooth driving), profitability (time to destination, fuel economy).
Environment	Roads, other traffic, pedestrians, customers, weather, city maps.
Actuators	Steering wheel, accelerator, brakes, horn, gear shift, display/voice for communicating with customers.
Sensors	Cameras (vision), GPS (location), speedometers, accelerometers, odometer, microphone (listening for customer requests), laser/radar (distance sensing).

Task environments

- Performance measure
  - How can we judge the automated driver?
  - Which factors are considered?
    - getting to the correct destination
    - minimizing fuel consumption
    - minimizing the trip time and/or cost
    - minimizing the violations of traffic laws
    - maximizing the safety and comfort, etc.

Task environments

- Environment
  - A taxi must deal with a variety of roads
  - Traffic lights, other vehicles, pedestrians, stray animals, road works, police cars, etc.
  - Interact with the customer

Task environments

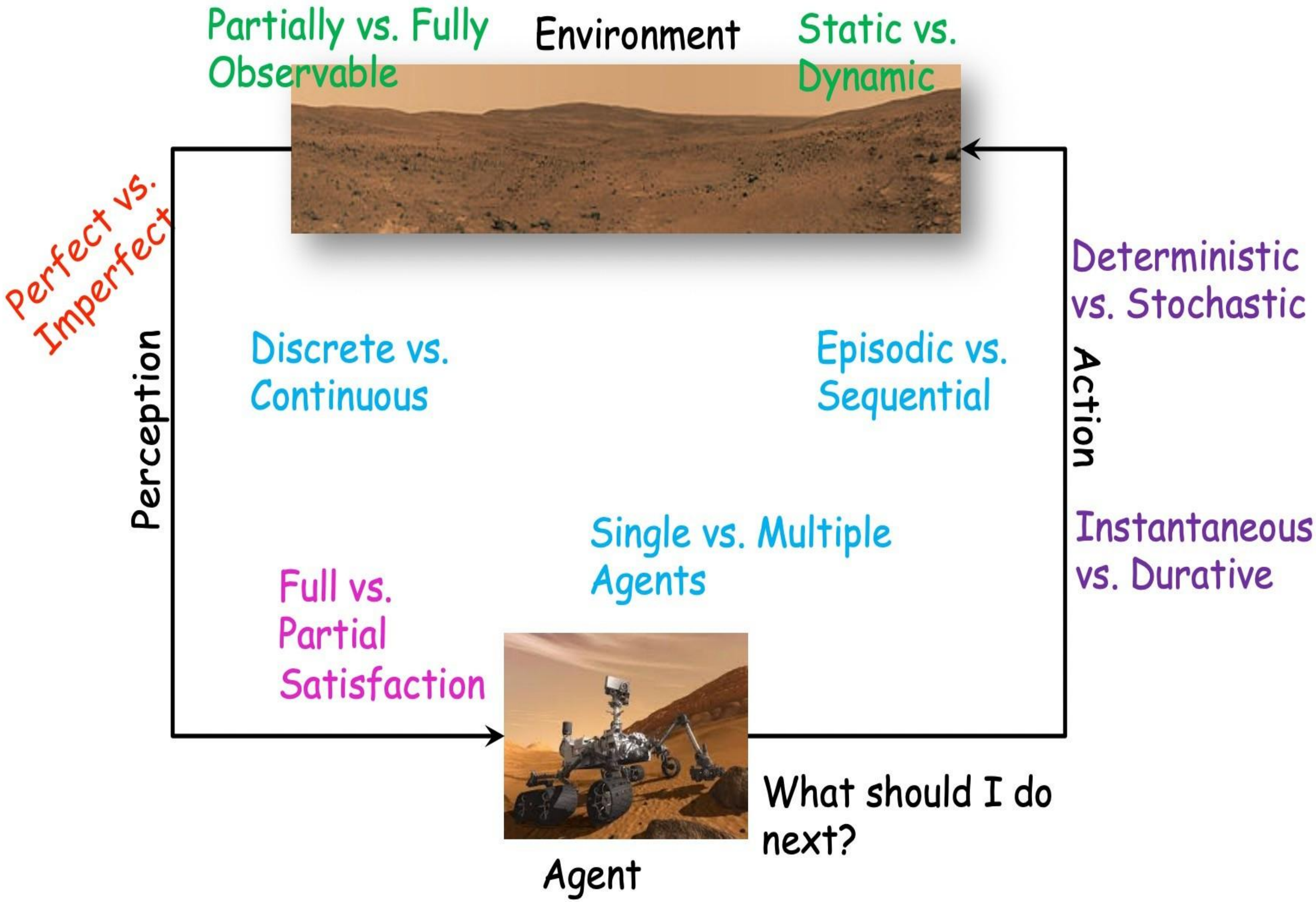
- Actuators (for outputs)
  - Control over the accelerator, steering, gear shifting and braking
  - A display to communicate with the customers
- Sensors (for inputs)
  - Detect other vehicles, road situations
  - GPS (Global Positioning System) to know where the taxi is
  - Many more devices are necessary

Task environments

- PEAS description of the task environment for an automated taxi driver.

Performance Measure	Environment	Actuators	Sensors
Safe, fast, legal, comfortable trip, maximize profits, minimize impact on other road users.	Roads, other traffic, police, pedestrians, customers, weather.	Steering, accelerator, brake, signal, horn, display, speech.	Cameras, radar, speedometer, GPS, engine sensors, accelerometer, microphones, touchscreen.

Properties of task environments



Properties of task environments

- **Fully observable** vs. **Partially observable**
  - **Fully observable**
    - An environment is fully observable if an agent's sensors **provide complete and accurate information** about the entire state of the environment at every point in time
    - Example: A chess game is fully observable since both players can see the entire board and all pieces at all times.
  - **Partially observable**
    - An environment is partially observable when the agent's sensors **provide incomplete, uncertain, or noisy information** about the environment's state. This may occur because the sensors are limited, inaccurate, or unable to capture some parts of the environment.
    - Example: A robot vacuum cleaner with a local dirt sensor cannot determine whether distant areas of a room are clean or dirty.

Properties of task environments

- **Deterministic** vs. **stochastic**
- **Deterministic Environment:**
  - The next state of the environment is completely determined by the current state and the action executed. The result is always predictable.
  - Example: Using a calculator. If the current state is the number '5' displayed, and the agent performs the action 'press + 3 =', the resulting state will always be '8'. There is no randomness or uncertainty in the outcome of the action.
- **Stochastic Environment:**
  - The agent cannot know the exact outcome of its action. That is, even with the same action and current state, different outcomes may occur.
  - Example: The vacuum cleaner and taxi driver agents operate in stochastic environments due to unobservable factors such as random dirt appearance, unpredictable passengers, or noisy sensors.

Properties of task environments

- **Episodic** vs. **Sequential Environments**
  - **Episodic Environment:**
    - Each episode consists of a single perception–action pair by the agent.
    - The outcome of one episode does not depend on previous episodes.
    - Each decision is made independently of past experiences.
    - Episodic environments are generally simpler because the agent doesn't need to plan ahead.
    - Example: An image classification system — each image is processed separately.
  - **Sequential Environment:**
    - The current action can influence future states and decisions.
    - The agent must look ahead and plan a sequence of actions.
    - Examples: Driving a taxi, playing chess.

Properties of task environments

- **Static** vs. **dynamic**
  - A dynamic environment is always changing over time
    - E.g., Driving a car in heavy traffic
  - In a static environment, the environment does not change while the agent is deciding on an action
    - E.g., Solving a Crossword puzzles
- **Semidynamic**
  - The environment itself does not change, but the agent's performance score (reward or penalty) changes over time.
    - E.g., Timed Quiz System—The questions stay the same (environment doesn't change); However, the score decreases as time passes.

Properties of task environments

Discrete vs. continuous

Discrete Environment:

- The environment consists of a finite number of several states, percepts, and actions. The agent's decisions are made in clearly defined steps.
- Example: Playing chess — the board has a limited number of positions and moves.

Continuous Environment:

- The environment contains an infinite range of possible states or actions.
- The agent operates in a smooth, real-world space that changes continuously over time.
- Example: Taxi driving — positions, speeds, and passenger behaviors vary continuously.

Properties of task environments

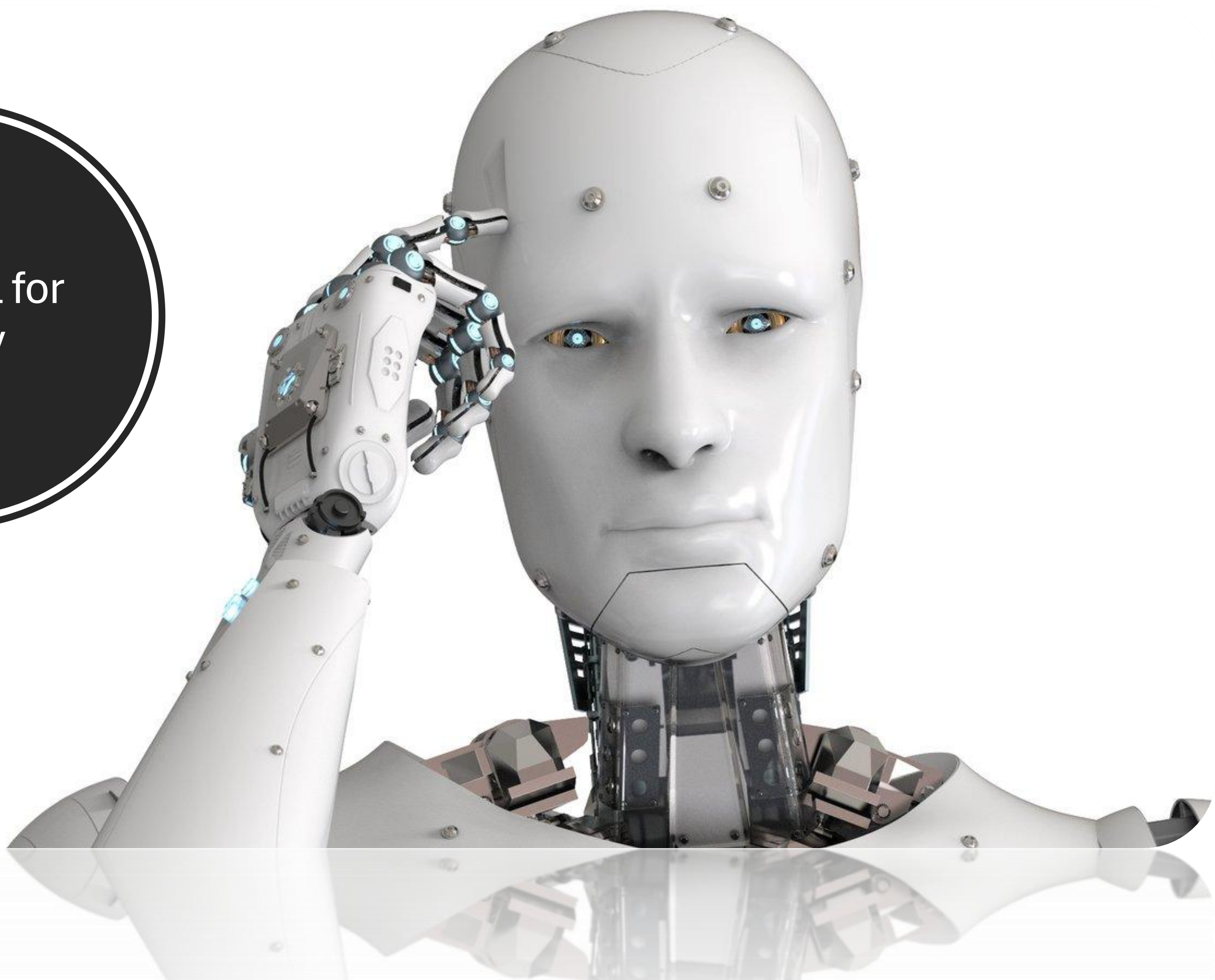
- **Single agent**
  - The environment contains only one agent acting and trying to achieve its goal.
  - Ex., Maze-solving robot
- **Multiagent**
  - A multi-agent environment involves two or more agents interacting. The success of one agent depends not only on its actions but also on others' actions.
  - Ex., Online multiplayer games

Properties of task environments

- **Known** vs. **unknown**
  - **Known**
    - The agent has complete knowledge of how the environment responds to all possible actions. The outcomes of every action are predictable and documented.
    - Ex. A Self-Driving Car has a perfect, pre-loaded digital map of a city. It knows the location of every street, traffic light, etc.. The outcome of taking any route is predictable.
  - **Unknown**
    - The agent lacks complete knowledge of the environment's dynamics and must discover the consequences of actions through interaction and learning.
    - Ex. The car is driving in a city right after a major earthquake. Roads are blocked, new temporary traffic patterns are in place, and bridges are out. The car's pre-existing map is now largely incorrect.

Examples of task environments

Task Environment	Observable	Agents	Deterministic	Episodic	Static	Discrete / Continuous
Crossword puzzle	Fully	Single	Deterministic	Sequential	Static	Discrete
Chess with a clock	Fully	Multi	Deterministic	Sequential	Semi	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
English tutor	Partially	Multi	Stochastic	Sequential	Dynamic	Discrete



That’s all for Today