Agents

Intelligent Systems II

Mário Antunes

February 11, 2026

Universidade de Aveiro

Agents and Intelligent Agents

What is an Agent?

- **Definition**: An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through actuators.
- **The Core Concept**: It is an entity that exercises control over its own actions to achieve goals.
- **Examples**:
  - **Biological**: Humans, animals.
  - **Robotic**: Robotic vacuum cleaners, assembly line arms.
  - **Software**: Web crawlers, NPCs in video games, Trading bots.

The Agent Interaction Loop i

The fundamental cycle of an agent involves four steps:

1. **Sense**: Gather data from the environment via sensors.
2. **Process/Decide**: The "Black Box" of intelligence processing the input.
3. **Act**: Execute an action via actuators.
4. **Alter**: The action changes the state of the environment.
    - *Repeat*: The agent senses the *new* state.

The Agent Interaction Loop ii

Analogy: A pilot flying a plane.

- *Sense*: Reads altimeter and looks out the window.
- *Decide*: “I am too low.”
- *Act*: Pulls the yoke back.
- *Alter*: The plane’s elevators move, air pressure changes, plane rises.

Sensors: Perceiving the World

- **Sensors** are the bridge from the physical/virtual world to the agent's internal representation.
- **Percept sequence**: The complete history of everything the agent has perceived.
  - $P = \{p_1, p_2, ..., p_t\}$
- **Types**:
  - *Passive*: Cameras, Microphones (Receive energy).
  - *Active*: Radar, Lidar, Ultrasound (Emit energy and measure return).
  - *Proprioceptive*: Measuring internal state (Battery level, wheel rotation).

Actuators: Affecting the World

- **Actuators** are the mechanisms that allow the agent to exert force or influence.
- **Mapping**: The agent function maps percepts to actions.
    - $f : P^* \to A$
- **Examples**:
    - *Physical*: Motors, hydraulic cylinders, speakers, screens.
    - *Virtual*: Sending a network packet, updating a database record, writing a file.

The Environment

The “World” in which the agent operates. It defines the constraints and the physics.

- **State (S)**: A snapshot of the environment at a specific time t.
- **Transition**: The environment evolves based on the agent’s actions and its own natural dynamics.
    - $S_{t+1} = T(S_t, A_t)$
- **Key Challenge**: The agent rarely sees the *entire* state (Partial Observability).

Dimensions of the Environment

To design an intelligent agent, we must classify the environment:

1. **Fully vs. Partially Observable**: Do sensors detect the full state? (Chess vs. Poker).
2. **Deterministic vs. Stochastic**: Is the next state determined purely by current state + action? (Crossword vs. Dice game).
3. **Static vs. Dynamic**: Does the world change while the agent is thinking? (Crossword vs. Tennis).
4. **Discrete vs. Continuous**: (Detailed in next slides).

Discrete vs. Continuous Worlds

This distinction fundamentally changes the math and algorithms used.

- **Discrete World**:
  - Finite number of distinct states.
  - Time moves in steps (turns).
  - *Math*: Sets, Graphs, State Machines.
- **Continuous World**:
  - States are defined by real numbers ($\mathbb{R}^n$).
  - Time flows smoothly.
  - *Math*: Calculus, Differential Equations, Physics engines.

Discrete vs. Continuous Actions

Just as the world can be granular or smooth, so can the agent's choices.

- **Discrete Actions**:
  - Finite set of choices. e.g., $\{Up, Down, Left, Right\}$.
- **Continuous Actions**:
  - Parameters usually range over continuous values. e.g., Steering angle $\theta \in [0, 360]$, Acceleration $a \in [0, 100]$.

World State Definitions

We need more precise terms to describe the complexity of the state space.

Term	Definition	Example
Finite / Tabular	A limited, countable number of states.	Tic-Tac-Toe, Grid World
Infinite / High-Dimensional	Uncountable states. Requires function approximation.	Robot Arm Joint Angles ($\mathbb{R}^n$)
Fully Observable	The agent knows the exact state S_t.	Chess, Video Games (HUD)
Partially Observable (POMDP)	The agent only knows a probability distribution.	Poker, Fog of War, Real-world
Structured	State has a known schema/format.	Database Record, JSON
Unstructured / Sub-symbolic	Raw data where features must be extracted.	Pixel data, Audio waveform

Action Type Definitions

Similarly, actions can be more nuanced than just a simple binary choice.

Term	Definition	Example
Categorical (Discrete)	Selecting one option from a distinct list.	`Press {A, B, X, Y}`
Parametric (Continuous)	Controlling a value on a sliding scale.	`Steer(34.5°)`
Hybrid / Parameterized	Choosing a category AND a parameter.	`Pass_Ball(Angle=45, Power=0.8)`
Deterministic	Action A always leads to State S'.	`Move(Right)` $\rightarrow (x+1, y)$
Stochastic	Action A leads to S' with probability P.	`Shoot(Target)` $\rightarrow$ Hit (80%) / Miss (20%)
Primitive vs. Macro	Atomic actions vs. High-level sequences.	`Motor_Voltage(5V)` vs. `Go_To_Room(B)`

Actions vs. World State: A Taxonomy

Combining these dimensions gives us four distinct types of agent environments.

	Discrete Action	Continuous Action
Discrete World	**Chess / Tic-Tac-Toe** *State:* Finite board positions *Action:* Move piece to square (e4)	**Ant Colony Optimization** *State:* Graph nodes (cities) *Action:* Deposit pheromone intensity $I \in [0, 1]$
Continuous World	**Video Games (Platformers)** *State:* (x, y, z) coordinates *Action:* Press Jump Button (0 or 1)	**Autonomous Driving / Robotics** *State:* Physics, Friction, Velocity *Action:* Steering angle θ, Acceleration a

Case Study 1: Discrete World / Discrete Actions

- **Example**: **Tic-Tac-Toe** or **Chess**.
- **World**: A 3x3 grid (finite states).
- **Action**: Place 'X' in cell (i, j).
- **Why it's "Easy"**: We can often enumerate all possibilities (Search Trees).
- **Analogy**: Multiple Choice Question test. You select A, B, C, or D. There is no "Choice A.5".

Case Study 2: Continuous World / Discrete Actions

- **Example**: **Modern Video Games** (e.g., a platformer or FPS).
- **World**: Continuous coordinates (x, y, z) represented by floats. Objects move smoothly.
- **Action**: Buttons on a controller. You press “Jump” or you don’t.
- **Complexity**: The agent must decide *when* to trigger a discrete action in a flowing world.
- **Analogy**: Taking a photo of a race car. The car moves continuously, but the shutter click is a single, discrete event.

Case Study 3: Continuous World / Continuous Actions

- **Example**: **Autonomous Driving** or **Robotic Arms**.
- **World**: The road, physics, friction (Real numbers).
- **Action**:
    - Steering wheel: $34.5°$
    - Brake pressure: 0.82 Bar
- **Complexity**: Infinite search space. Requires control theory or regression-based learning.
- **Analogy**: Pouring water into a glass. You adjust the tilt continuously to control the flow rate.

From “Agent” to “Intelligent Agent”

An agent becomes **intelligent** based on how it implements the “Process/Decide” step.

The Spectrum of Intelligence Sources:

1. **Symbolic/Logic**: Explicit rules and reasoning.
2. **Planning**: Searching for sequences of actions.
3. **Machine Learning**: Learning from data (probabilities).
4. **Deep Learning**: Learning representations.
5. **Agentic AI**: Generative models (LLMs).

Approach A: Logic and Rule Engines

- **Basis**: First-Order Logic (FOL) and Propositional Logic.
- **Mechanism**: The agent has a Knowledge Base (KB) of facts and rules.
- **Process**:
    1. Sense $\rightarrow$ Convert to Logical Fact (`dirt(loc_A)`).
    2. Query KB $\rightarrow$ `?- action(X).`
    3. Inference $\rightarrow$ Resolution/Modus Ponens derives `X = suck`.
- **Pros**: Explainable, verifiable.
- **Cons**: Brittle in noisy/unknown environments.

The Prolog Agent

We can use **Prolog** to implement this logic.

- **World Model**: Defined by predicates (`at(agent, 1, 1)`, `pit(2, 2)`).
- **Reasoning**: `prolog move(forward) :- clear_ahead, \+ pit_ahead. move(turn_left) :- wall_ahead.`
- **Logic Revision**: We use Horn Clauses and Resolution to prove which action leads to a goal.

Approach B: Planning Agents

- **Basis**: Search Algorithms (A*, BFS) and STRIPS operators.
- **Mechanism**: The agent has a model of “If I do X, Y happens.”
- **Process**:
  - *Goal*: `Holding(BlockA)`
  - *Plan*: Look ahead into the future to find a sequence: `MoveTo(A) -> PickUp(A)`.

Approach C: Machine Learning (RL)

- **Basis**: Inductive Inference (Generalizing from examples).
- **Mechanism**: Reinforcement Learning (RL).
- **Feedback Loop**:
    - Agent acts $\rightarrow$ Environment responds with **Reward** (R).
    - Agent updates its policy (π) to maximize cumulative reward.
- **Analogy**: Dog training. Treat = Reward; Scolding = Punishment. The dog learns “Sit” = “Treat”.

Approach D: Deep Learning Agents

- **The Shift**: Traditional RL struggled with raw inputs (pixels).
- **Deep Q-Networks (DQN)**: Use a Neural Network to estimate the value of actions directly from sensors (e.g., video frames).
- **Capability**: Can play Atari games, Drive cars, fold proteins.

Approach E: Agentic AI (GenAI & LLMs)

- **The New Frontier**: Using Large Language Models (LLMs) as the reasoning engine.
- **Mechanism (ReAct Pattern)**:
  1. **Thought**: LLM analyzes context. “I need to find the file size.”
  2. **Action**: LLM outputs a tool call. `os.stat('file.txt')`
  3. **Observation**: The tool returns output. “2048 bytes”.
  4. **Repeat**.
- **Advantage**: Handles unstructured natural language and “common sense” reasoning better than strict logic.

Summary of Intelligence Architectures

Architecture	Representation	Reasoning Method	Best For
Logic	Symbols/Facts	Theorem Proving	Defined constraints (Tax)
Planning	States/Actions	Search (A*)	Logistics, Robotics
RL/Deep	Vectors/Tensors	Optimization	Motor control, Games
GenAI	Text/Tokens	Probabilistic Generation	Creative/Office tasks

The Challenge of Continuous Worlds

How do we apply Logic or Tabular RL (like Q-Learning) to a continuous world?

- **Problem**: A Q-Table requires a row for every state.
- *Continuous*: Infinite states ($x = 1.001, x = 1.002...$).
- *Result*: The table would be infinite. Memory explosion.

Solution 1: Discretization (Bucketing)

Turn the continuous world into a discrete one.

- **Technique**: Divide the continuous range into “buckets” or grid cells.
- **Example**:
  - Real Temp: $22.4°C \rightarrow$ Bucket: “Warm” ($20 - 25°C$).
  - Real Dist: $5.67m \rightarrow$ Grid Cell: $(5, 6)$.
- **Issue**: **Curse of Dimensionality**. If you have 10 dimensions and divide each into 100 buckets, you need 100^{10} states!

Solution 2: Function Approximation

Instead of remembering every state, learn a **function** that estimates the value.

- **Logic**: Instead of a lookup table $Q(s, a)$, use a function $Q(s, a; \theta) \approx f(s)$.
- **Linear**: $Value = w_1 \cdot x + w_2 \cdot y + b$.
- **Non-Linear**: Neural Networks (Deep Learning).
- *Analogy*: You don't memorize the answer to 234×567; you learn the *algorithm* of multiplication to handle any numbers.

Handling Continuous Actions

Choosing one action from infinite possibilities.

- **Discretization**: Limit steering to $\{-10^\circ, 0^\circ, +10^\circ\}$. (Jerky movement).
- **Policy Gradient Methods**:
  - Instead of outputting a specific value, the network outputs the **parameters of a probability distribution** (Mean μ and Standard Deviation σ).
  - Action $a \sim \mathcal{N}(\mu, \sigma)$.
- **Actor-Critic**: The “Actor” proposes a continuous action, the “Critic” estimates its value.

Analogy: The Dart Thrower

- **Task**: Throw a dart at the center (Bullseye).
- **World**: Continuous (Physical space).
- **Action**: Continuous (Angle of arm, speed of release).
- **Learning**:
    - *Discrete approach*: Divide the board into squares. "I hit square A4".
    - *Continuous approach*: "I missed by 2cm left." Adjust muscle tension slightly (Gradient Descent).

Agent Types: Simple Reflex Agent

- **Mechanism**: Condition-Action Rules.
- **Logic**: `if (Condition) then (Action)`.
- **Memory**: None. Ignores history.
- **Example**: Thermostat.
  - `if (Temp < 20) then (Turn_On_Heater)`.
- **Limitation**: Fails if the environment is partially observable (needs memory to know what is unobserved).

Agent Types: Model-Based Reflex Agent

- **Mechanism**: Keeps internal state to track the world.
- **Logic**: `State = Update(OldState, Action, Percept)`.
- **Example**: Self-driving car waiting at a red light.
    - If a truck blocks the camera, the car remembers “The light was red 2 seconds ago” even if it can’t see it now.
- **Key Component**: “Model of the World” (Physics/Logic).

Agent Types: Goal-Based Agent

- **Mechanism**: Acts to achieve a specific desirable state.
- **Logic**: Search and Planning. “Is action A leading me closer to Goal G?”
- **Difference**: Reflex agents respond to the *past/present*. Goal agents look to the *future*.
- **Example**: GPS Navigation.
  - Goal: “Home”.
  - Action: Turn left (because it reduces distance to home, not just because the road turns).

Agent Types: Utility-Based Agent

- **Mechanism**: Maximizes a “Happiness” function (Utility).
- **Nuance**: Goals are binary (Achieved/Not Achieved). Utility is continuous.
- **Example**: Taxi routing.
  - Goal: Get to destination.
  - Utility: Get to destination *quickly* AND *safely* AND *save fuel*.
  - Logic: Choose action that maximizes $E[Utility]$.

Recap:

We explore specific implementations of these agents:

1. **Representation**: Prolog/Knowledge Graphs (Slide 13).
2. **Communication**: NLP/Grammars (Slide 4).
3. **Learning**:
   - **RL**: Q-Learning/Deep Q-Learning for dynamic worlds (Slide 17).
   - **ILP**: Inductive Logic Programming to learn rules from data (Slide 16).

Conclusion

Modern AI often combines these approaches
(Neuro-Symbolic AI).

- **Sensing**: Deep Learning (Vision).
- **Reasoning**: Logic/Planning (Strategy).
- **Action**: Control Theory (Movement).

Final Thought: An intelligent agent is not just an algorithm; it is a system that exists in a loop with its reality.