AI Assistants: Alexa and Siri – Examples of Intelligent Agents

AI assistants like **Alexa** and **Siri** function as **intelligent agents** by processing user requests and delivering responses based on vast datasets.

How They Work

- **Sensors & Inputs**: Use **microphones and other sensors** to perceive user requests.
- **Decision-Making**: Analyze requests by accessing **supercomputers and data banks worldwide**.
- **Internet-Based Knowledge**: Pull information from the web **without user intervention**.

Rational Agents

Definition

A **Rational Agent** is an **entity (person, machine, firm, or software)** that **makes decisions** to maximize the best possible outcome based on:

1. **Past percepts** (previous experiences)
2. **Current percepts** (present input from the environment)

AI System as a Rational Agent

- **AI System = Agent + Environment**
- **Agent**: Performs actions.
- **Environment**: Includes external factors and possibly other agents.

Agent Components

1. Perception (Sensors)

- Agents **perceive** their environment through **sensors** (e.g., microphones, cameras, GPS).

2. Actions (Actuators)

- Agents act upon their environment through actuators (e.g., speakers, displays, robotic arms).

Agent Architecture

- Architecture = Sensors + Actuators
- Agent = Architecture + Program
  - Architecture: The physical structure (e.g., a smartphone or smart speaker).
  - Program: The AI logic that processes inputs and makes decisions.

Example: Alexa

- Sensors: Microphone for voice input.
- Actuators: Speaker for audio response.
- Program: AI model that interprets speech and executes tasks.

Agents and Environments

An agent is any entity that perceives its environment through sensors and acts upon it using actuators.

Types of Agents

1. Human Agent

- Sensors: Eyes, ears, and other sensory organs.
- Actuators: Hands, legs, vocal tract, etc.

2. Robotic Agent

- Sensors: Cameras, infrared range finders.
- Actuators: Motors, robotic arms, wheels.

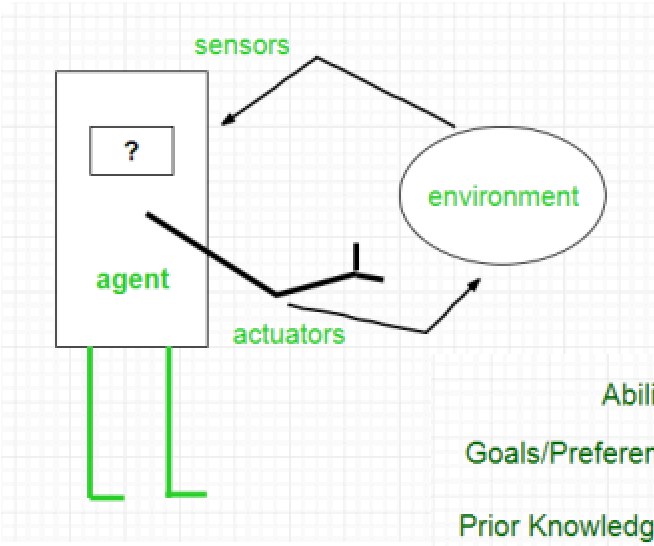
3. Software Agent

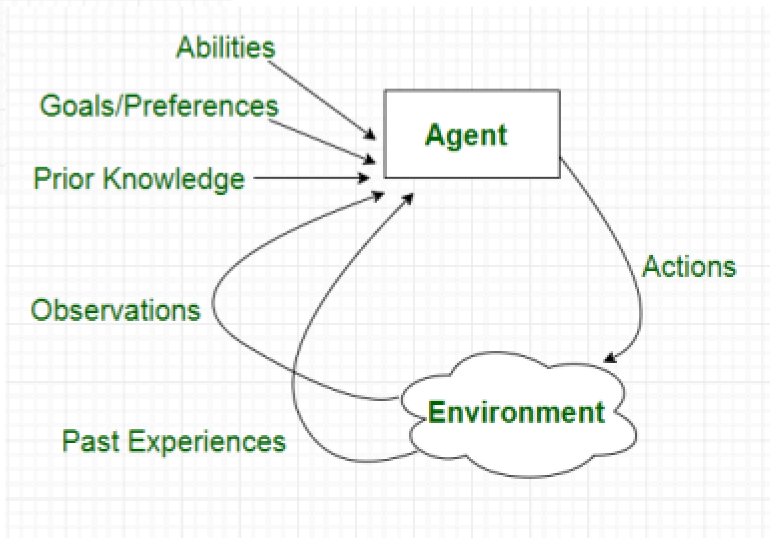
- Sensors: Keystrokes, file contents, network packets.

- **Actuators**: Displaying information on the screen, writing files, sending network packets.

Agent-Environment Interaction

- Agents **perceive** the environment using **sensors**.
- Agents **act** upon the environment using **actuators**.
- The **type of agent** depends on the **nature of the environment** and the **tasks it performs**.





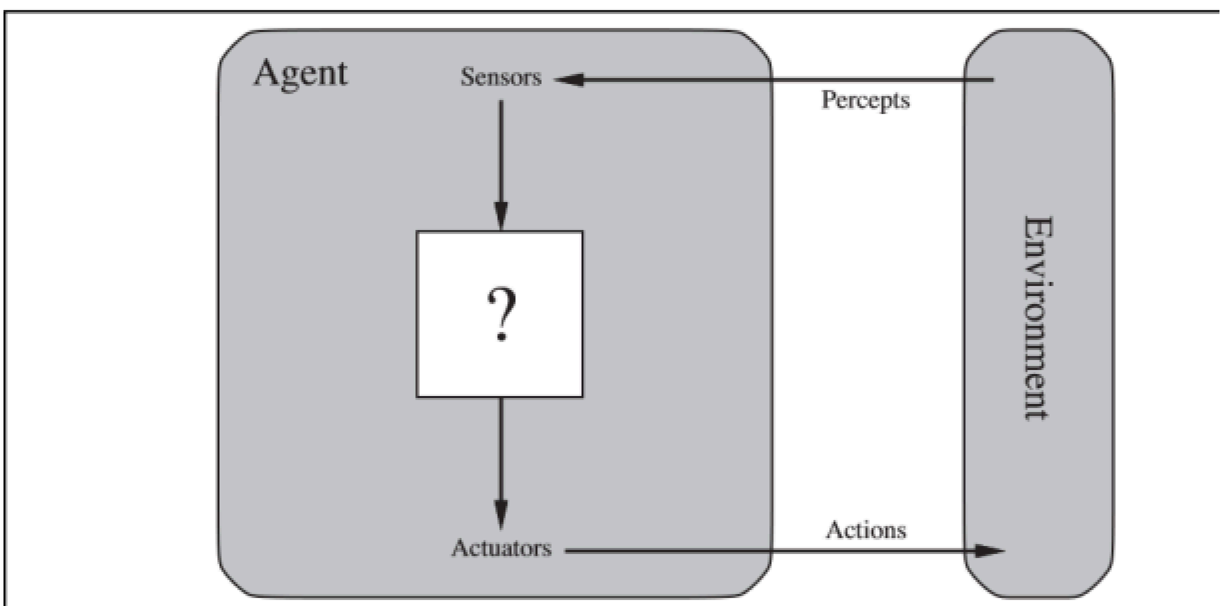


Figure 2.1 Agents interact with environments through sensors and actuators.

Percept, Agent Function, and Agent Program

1. Percept

- **Definition**: An agent's **perceptual inputs** at any given instant.

- **Percept Sequence**: The complete **history** of everything the agent has ever perceived.
- **Decision Making**:
  - An agent's **choice of action** depends on the **entire percept sequence** observed so far.
  - The agent **cannot** act based on information it **hasn't perceived**.

2. Agent Function

- **Definition**: A mapping from **percept sequences** to **actions**.
- **Determines**:
  - How the agent **chooses an action** based on what it has perceived.

3. Agent Program

- **Definition**: The implementation of the **agent function** in an **artificial agent**.
- **Executes**:
  - The logic that determines the **agent's behavior** based on percept sequences.

Classic Vacuum Cleaner Problem

1. Overview

- **Vacuum cleaner problem** is a classic **search problem** in AI.
- The **vacuum cleaner** is the **agent**.
- The **goal** is to **clean** all rooms.

2. Environment

- Two rooms (**Room A** and **Room B**).
- Each room may contain **dirt**.
- The vacuum cleaner starts in **either room**.

3. Operations

- **Move Left**: Move to the **left** room.
- **Move Right**: Move to the **right** room.
- **Suck Dirt**: Remove **dirt** from the current room.

4. Goal State

- **Both rooms** must be **clean** and dust-free.

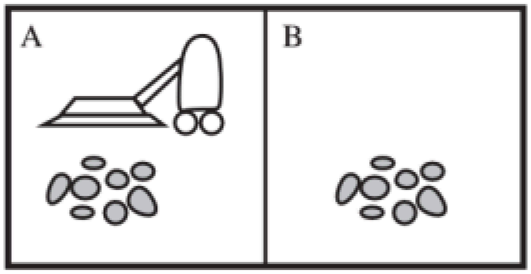


Figure 2.2 A vacuum-cleaner world with just two locations.

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
⋮	⋮

Figure 2.3 Partial tabulation of a simple agent function for the vacuum-cleaner world shown in Figure 2.2.

Rationality of an Agent

Rationality of an **agent** depends on four factors:

1. **Performance Measure**: Defines the **success criteria**.
2. **Prior Knowledge**: The agent's **understanding** of the **environment**.
3. **Actions**: The set of **actions** the agent can perform.
4. **Percept Sequence**: The history of **percepts** observed by the agent.

Definition

- **Percept Sequence**: The **series of observations** an agent has made.
- **Rational Agent**: Selects an **action** that **maximizes performance** based on:
  - **Percept sequence**
  - **Built-in knowledge**
  - **Available actions**

Rationality of an Agent

An agent's rationality depends on the following four factors:

1. Performance Measure

- Defines the criterion for the agent’s success.
- Helps evaluate how well the agent is performing in its environment.

2. Prior Knowledge of the Environment

- The agent's understanding of its surroundings before it starts operating.
- More prior knowledge helps the agent make better decisions.

3. Actions the Agent Can Perform

- The set of possible actions available to the agent.
- Determines the agent’s ability to interact with and influence the environment.

4. Percept Sequence to Date

- **Definition:** The sequence of all percepts (inputs) the agent has received so far.
- The agent uses the percept sequence and built-in knowledge to decide the next action that maximizes performance.

PEAS (Performance Measures, Environment, Actuators, Sensors)

PEAS is a framework used to define AI agents by specifying:

1. Performance Measures

- Evaluates how well the agent is achieving its goal.

2. Environment

- The real-world context in which the agent operates.

3. Actuators

- The components that allow the agent to take actions.

4. Sensors

- The components that allow the agent to perceive the environment.

Example: PEAS for an Automated Car Driver

Category	Description
Performance Measures	- **Safety:** Avoid accidents and follow traffic rules. - **Optimal Speed:** Maintain an appropriate speed for different conditions. - **Comfort:** Provide a smooth driving experience.
Environment	- **Roads:** City streets, highways, rural roads, etc. - **Traffic Conditions:** Varying traffic densities and movement patterns.
Actuators	- **Steering Wheel:** Controls direction. - **Accelerator, Brake, and Gear:** Controls speed and movement.
Sensors	- **Cameras:** Detect road, obstacles, and traffic signs. - **GPS:** Determines the car's location. - **Speedometer, Odometer:** Measures speed and distance. - **Accelerometer:** Detects acceleration changes. - **Sonar:** Detects nearby objects.

PEAS Model Breakdown

- **Performance:** Defines the required qualities of the agent.
- **Environment:** Specifies where the agent operates.
- **Actuators:** Determines how the agent interacts with the environment.
- **Sensors:** Determines how the agent perceives the environment.

PEAS Descriptions for Different AI Agents

Below are PEAS (Performance, Environment, Actuators, Sensors) descriptions for various AI agents:

---

1. Mathematician's Theorem-Proving Assistant

Category	Description
Performance Measure (P)	- Possesses strong mathematical knowledge. - Can prove theorems accurately. - Minimizes steps and time required for proofs.
Environment (E)	- Internet for accessing online mathematical resources. - Library for reference materials.
Actuators (A)	- Display to present solutions and proofs.
Sensors (S)	- Keyboard for receiving input from the mathematician.

---

2. Autonomous Mars Rover

Category	Description
Performance Measure (P)	- Successfully explores and reports on terrain. - Collects and analyzes samples effectively.
Environment (E)	- Launch vehicle (during travel). - Lander (for descent and initial setup). - Mars (as the primary exploration environment).
Actuators (A)	- Wheels/legs for movement. - Sample collection device for gathering materials. - Analysis devices for examining samples. - Radio transmitter for communication with Earth.
Sensors (S)	- Camera for capturing images and navigation. - Touch sensors for detecting physical interactions. - Accelerometers and orientation sensors for stability. - Wheel/joint encoders for tracking movement. - Radio receiver for receiving commands from Earth.

---

3. Internet Book-Shopping Agent

Category	Description
Performance Measure (P)	- Finds and obtains requested or interesting books. - Minimizes expenditure while purchasing books.
Environment (E)	- Internet as the platform for book searching and purchasing.
Actuators (A)	- Follows links to navigate websites. - Enters and submits data in fields for purchasing. - Displays relevant book options to the

Category	Description
	user.
Sensors (S)	- Web pages for gathering book information. - User requests as inputs for search queries.

---

4. Robot Soccer Player

Category	Description
Performance Measure (P)	- Maximizes chances of winning the game. - Focuses on scoring goals and preventing opponent goals.
Environment (E)	- Soccer field as the playing area. - Ball as the object to be controlled. - Own team for coordination. - Opposing team as competitors. - Own body for movement and action execution.
Actuators (A)	- Legs or other locomotion devices for movement. - Kicking mechanism for ball control.
Sensors (S)	- Camera for visual perception. - Touch sensors for detecting contact with the ball. - Accelerometers for tracking speed and movement. - Orientation sensors for balance and position awareness. - Wheel/joint encoders for precise motion control.

Agent Type	Performance Measure	Environment	Actuators	Sensors
Medical diagnosis system	Healthy patient, reduced costs	Patient, hospital, staff	Display of questions, tests, diagnoses, treatments, referrals	Keyboard entry of symptoms, findings, patient's answers
Satellite image analysis system	Correct image categorization	Downlink from orbiting satellite	Display of scene categorization	Color pixel arrays
Part-picking robot	Percentage of parts in correct bins	Conveyor belt with parts; bins	Jointed arm and hand	Camera, joint angle sensors
Refinery controller	Purity, yield, safety	Refinery, operators	Valves, pumps, heaters, displays	Temperature, pressure, chemical sensors
Interactive English tutor	Student's score on test	Set of students, testing agency	Display of exercises, suggestions, corrections	Keyboard entry

Figure 2.5 Examples of agent types and their PEAS descriptions.

Omniscience, Learning, and Autonomy

1. Omniscient Agent

- **Definition:** An agent that knows the actual outcome of all its actions and can make perfect decisions.
- **Reality Check:** True omniscience is impossible because:
  - The real world is unpredictable.
  - No agent can have complete knowledge of all external factors.

Example: Crossing the Street

- A person rationally decides to cross the street when no immediate danger is visible.
- Unexpectedly, a cargo door falls from a plane, causing an accident.
- **Key Insight:** The person acted rationally based on available information but was not omniscient.
- **Conclusion:** Lack of omniscience does not imply irrationality.

2. Learning in AI

- AI systems can improve performance over time by analyzing past experiences.
- Example: **Sophia (AI Robot by Hanson Robotics)**
    - Designed to engage in human conversations.
    - Uses conversation data to refine responses.
    - Purpose: Assist the elderly in nursing homes and provide crowd interaction.

---

3. Autonomy in AI

- **Definition:** An agent's ability to operate without human intervention.
- **Relation to Learning:**
    - An AI becomes more autonomous as it learns from its environment.
    - Example: Sophia adapts its responses based on interactions.

Here's the detailed version of your notes while maintaining all points and proper formatting:

Task Environment

A task environment refers to the **choices, actions, and outcomes** available to an agent for a given task.

Properties of Task Environments:

Fully Observable vs. Partially Observable

- **Fully Observable:** The agent's sensors provide complete access to the environment's state at each point in time.
    - This makes decision-making easier as the agent does not need to maintain an internal state to track the world's history.
- **Partially Observable:** The agent has limited or noisy sensor data, making it difficult to determine the complete state of the environment.
- **Unobservable Environment:** If an agent has no sensors, it cannot perceive the environment at all.

Reasons for Partial Observability:

- **Noisy or inaccurate sensors** affect perception.
- **Some parts of the state are missing** from sensor data.

Examples:

- **Vacuum agent** – A local dirt sensor cannot detect dirt in other squares.
- **Automated taxi** – Cannot perceive what other drivers are thinking.

Comparison:

- **Fully Observable:** Chess – The board and opponent's moves are completely visible.
- **Partially Observable:** Driving – The agent cannot see around bends or detect hidden obstacles.

Single Agent vs. Multi-Agent

- **Single-Agent Environment:** Only one agent operates independently without interaction with other agents.
  - Example: A crossword-solving agent operates in a single-agent environment.
- **Multi-Agent Environment:** Multiple agents interact within the environment, which can be either **competitive** or **cooperative**.
  - **Competitive Multi-Agent:** Agents have conflicting goals. Example: Chess, where two agents compete.
  - **Cooperative Multi-Agent:** Agents work towards a shared or mutually beneficial outcome. Example: Taxi-driving, where avoiding collisions benefits all agents.

Deterministic vs. Stochastic

- **Deterministic Environment:** The next state is **completely determined** by the current state and the agent's action.
  - Example: **Vacuum world** (in its basic form) is deterministic—cleaning actions have predictable results.
- **Stochastic Environment:** The next state is **not fully predictable** due to randomness or hidden factors.
  - Example: **Taxi driving** is stochastic—traffic behavior is unpredictable, and unexpected failures (e.g., tire blowout) can occur.
  - Even the **vacuum world** can become stochastic if dirt appears randomly or if the suction mechanism fails.

Episodic vs. Sequential

- **Episodic Environment:** The agent's experience is divided into **independent episodes** where each action is based only on the current percept.
  - Example: **Defect detection on an assembly line**—each part is inspected independently, and decisions do not affect future parts.
- **Sequential Environment:** The current decision **affects future outcomes**, requiring long-term planning.
  - Example: **Chess and taxi driving**—a single move or driving decision can have long-term consequences.
- **Comparison:** Episodic environments are **simpler** because the agent does not need to **consider past or future states** when making decisions.

Static vs. Dynamic

- **Static Environment:** The environment **remains unchanged** while the agent is making a decision.
  - **Easier to handle** since the agent does not need to monitor changes continuously.
  - **Example: Mowing a lawn**—the grass does not change while the agent is mowing.
- **Dynamic Environment:** The environment **changes over time**, even while the agent is deliberating.
  - **Challenging** as the agent must **continuously observe and adapt** to changes.
  - **Example: Playing football**—other players move dynamically, affecting the game state.
- **Key Difference:** In a dynamic environment, **time plays a crucial role**, while in a static one, the agent can take time to decide without external changes.

Discrete vs. Continuous

- **Discrete Environment:**
  - **Has a finite number of distinct states, actions, and percepts.**
  - **Time is also handled in discrete steps.**
  - **Example: Chess**—the board has a **finite** number of positions, and moves are discrete.
- **Continuous Environment:**
  - **States, time, percepts, and actions vary smoothly.**
  - **Requires real-time processing and adaptation.**
  - **Example: Taxi driving**—speed, location, and steering angles change **continuously** over time.

- **Key Difference:** Discrete environments are **simpler to model** with a limited number of choices, while continuous environments require **handling infinite variations** dynamically.

Known vs. Unknown Environments

- **Known Environment:**
  - **The outcomes of all actions are provided.**
  - **The agent does not need to learn how the environment works.**
  - **Example: Tic Tac Toe**—rules and outcomes are predefined.
- **Unknown Environment:**
  - **The agent must learn the effects of actions through experience.**
  - **Example: A robot on Mars**—unknown terrain and physics require learning.

Comparison with Other Properties:

- **Fully/Partially Observable:** Is the state **completely known** or only **partially accessible**?
- **Deterministic/Stochastic:** Can the **next state be predicted** with certainty?
- **Episodic/Sequential:** Are actions **independent episodes** or do they **affect future decisions**?
- **Static/Dynamic:** Does the environment **change over time**?
- **Discrete/Continuous:** Are percepts/actions **limited** or **infinitely variable**?
- **Single/Multi-Agent:** Does the agent **act alone** or **with others**?

Examples of Different Environments:

- **Non-deterministic: Robot on Mars** (unpredictable physics)
- **Deterministic: Tic Tac Toe** (fixed rules)
- **Episodic: Mail sorting system** (each action is independent)
- **Non-episodic: Chess game** (moves impact future outcomes)
- **Discrete: Chess game** (finite moves)
- **Continuous: Driving a car** (infinite steering adjustments)

Four Basic Kinds of Agent Programs

1. Simple Reflex Agents

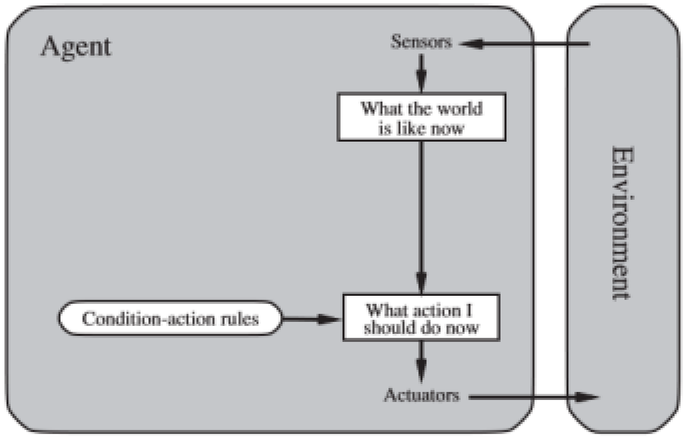


Figure 2.9 Schematic diagram of a simple reflex agent.

Simple Reflex Agents

- **Simplest type of agent**, selects actions based **only on the current percept**, **ignoring percept history**.
- Example:
  - A **vacuum agent** determines actions based **only on location and dirt presence**.
  - A **car braking system** reacts immediately when the **car in front brakes**.

Condition-Action Rule

- Defined as:
  - **if car-in-front-is-braking then initiate-braking**.
- Reflex actions are **quick responses** based **only on the current situation**.

Functions in Simple Reflex Agents

- **INTERPRET-INPUT**: Extracts an **abstract description** of the **current state** from percepts.
- **RULE-MATCH**: Finds the **first matching rule** from a set of predefined rules based on the **current state**.

Problems with Simple Reflex Agents

- **Limited intelligence**, as they rely only on **immediate percepts**.
- **Lack of memory** about **non-perceptual aspects** of the state.
- **Rigid rules** require **manual updates** whenever the **environment changes**.

Examples of Simple Reflex Agents

- **Thermostats** turning heaters on/off based on **current temperature**.

- Automatic doors opening when motion is detected.
- Traffic lights responding to vehicle presence sensors.

2. Model-Based Reflex Agents

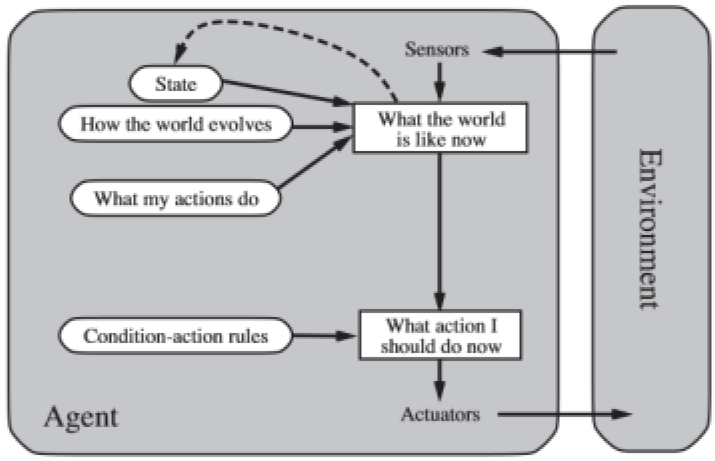


Figure 2.11 A model-based reflex agent.

Handling Partial Observability

- Agents must track unobservable parts of the world by maintaining an internal state.
- This internal state is based on percept history and helps reflect hidden aspects of the environment.

Updating Internal State Information

- Requires two types of knowledge:
  - How the world evolves independently of the agent
    - Example: An overtaking car will likely be closer behind than before.
  - How the agent's actions impact the world
    - Example: Turning the steering wheel clockwise makes the car turn right. Driving five minutes northbound results in being five miles further north.

Model-Based Agents

- A Model of the World encodes this knowledge, whether through simple Boolean circuits or complex scientific models.
- Agents that utilize such models are known as Model-Based Agents.

Example: Waymo

- Waymo, initially Google's autonomous car project (2009), is now a public trial self-driving system in Phoenix, Arizona.

- Functions as an **intelligent model-based agent**, using **sensors** to observe the environment and **actuators** to respond accordingly.

Update-State Function

- **Responsible for generating the new internal state description** based on the latest percepts and historical data.
- **Challenges**: In a **partially observable environment**, it is **rarely possible** to determine the exact **current state**.

Additional Examples of Model-Based Agents

- **AI-powered assistants** predicting user needs based on **past interactions**.
- **Robotic vacuum cleaners** mapping out rooms to **optimize cleaning paths**.
- **Stock market prediction models** adjusting strategies based on **economic trends and news**.

3. Goal-Based Agents

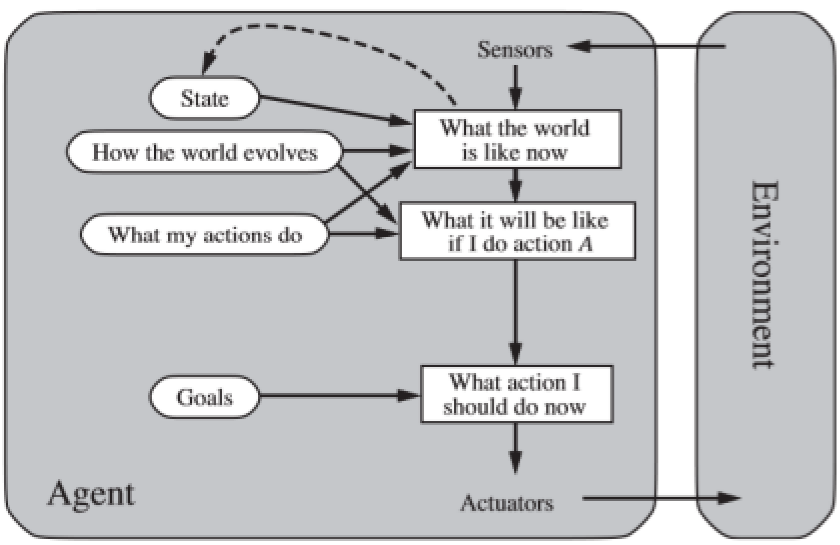


Figure 2.13 A model-based, goal-based agent. It keeps track of the world state as well as a set of goals it is trying to achieve, and chooses an action that will (eventually) lead to the achievement of its goals.

Goal-Based Agents

- **Knowing the current state of the environment is not always enough** to make decisions.
- Example: A **taxi at a junction** can **turn left, turn right, or go straight**, but the correct choice depends on **its destination**.

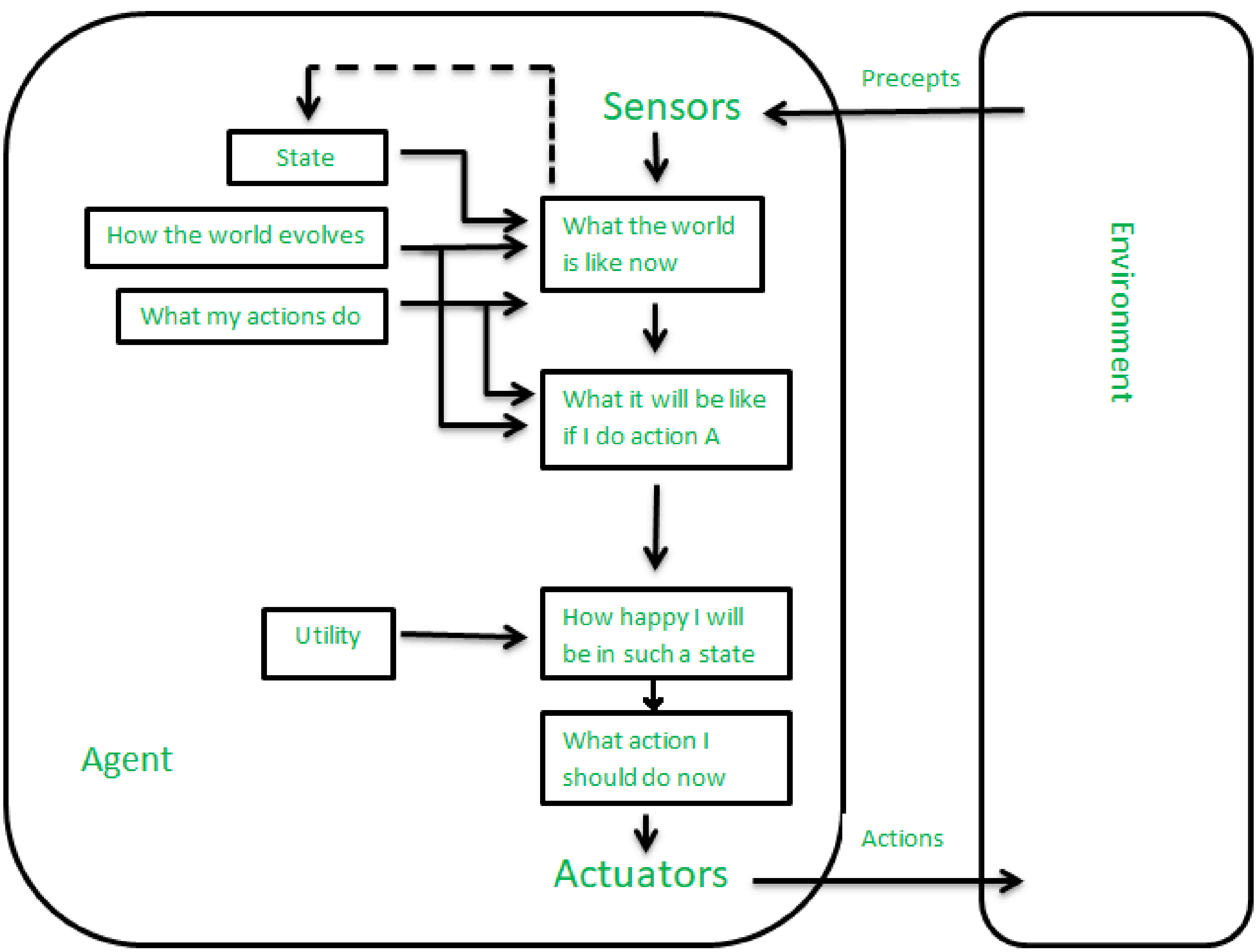
Goal-Driven Decision Making

- Agents must evaluate **what will happen if they take a certain action** and whether it **aligns with their goals**.
- Example:
  - A person making **New Year's resolutions** considers **whether a goal will bring long-term happiness**.
  - A navigation system selects **the best route** based on the **desired destination**.

Key Characteristics of Goal-Based Agents

- **Decisions are made based on future goals** rather than just **immediate conditions**.
- **More flexible** than reflex agents, as they consider **multiple actions before choosing the best one**.
- **Can plan ahead**, rather than reacting **only to the current situation**.

4. Utility-Based Agents



Utility-Based Agents

- **Goals provide only a binary distinction** between **"happy" and "unhappy"** states.
- Example: A taxi reaching its **destination** meets the goal, but **some routes** may be **quicker, safer, more reliable, or cheaper** than others.

Decision-Making Beyond Goals

- A **utility-based agent** does more than just **achieve a goal**—it evaluates the **quality of different solutions**.
- **Utility measures** help agents **compare and prioritize actions** based on factors like **efficiency, safety, and cost**.

Key Characteristics of Utility-Based Agents

- **Models and tracks its environment** using advanced techniques in **perception, reasoning, and learning**.
- **Optimizes outcomes** rather than just **meeting a goal**.
- **Evaluates multiple possible actions** to choose the **best** one for a given scenario.
- Example:
  - **Vacation Planning** – Choosing between **multiple destinations** based on **cost, convenience, and enjoyment**.
  - **Driving Decisions** – Selecting the **fastest and safest** route instead of just **reaching the destination**.

Learning Agents

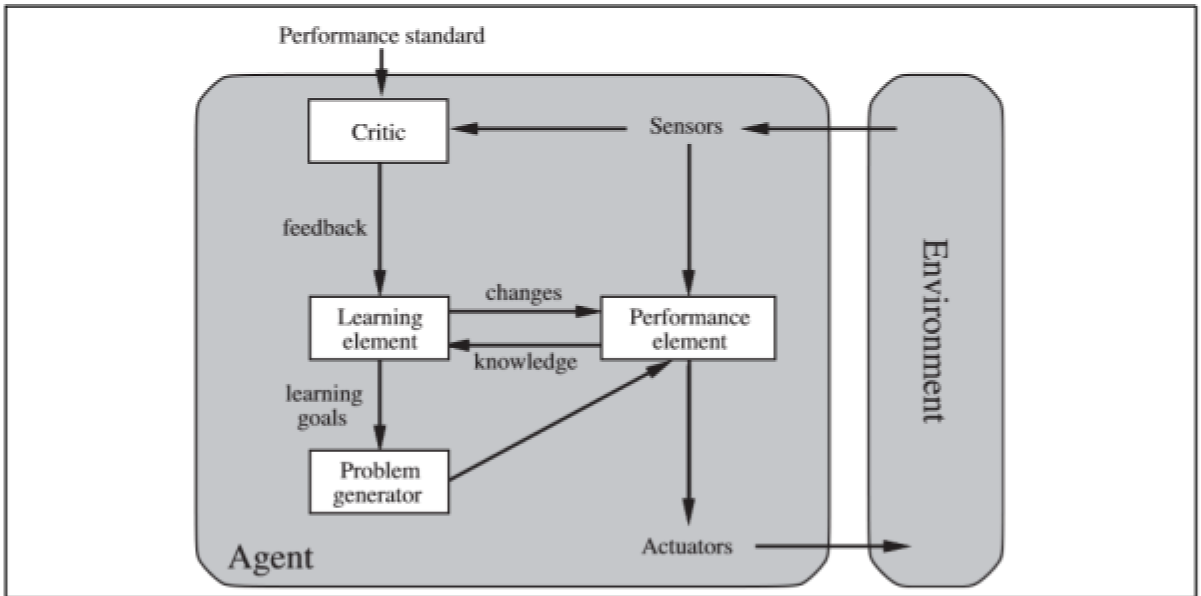


Figure 2.15 A general learning agent.

Learning Agents

- A **learning agent** is an AI tool capable of **improving its performance** based on **experience**.
- Starts with **basic knowledge** and **autonomously adapts** by learning from **past actions and feedback**.

Example: Learning Useful Information

- A **cab driver** usually takes the **Maple Street exit** but tries **Palace Road** for a change.
- **Palace Road** turns out to be **quicker and safer**, avoiding an **accident-prone intersection**.
- The driver **learns from this experience** and decides to **use Palace Road** in the future.

AI Perspective

- This example demonstrates how a **learning agent observes outcomes**, **analyzes effectiveness**, and **adapts decisions** over time.
- Learning agents are **crucial in AI** for **self-improvement, decision-making, and optimization** in various applications.

Components of a Learning Agent

A **learning agent** consists of **four conceptual components**, each playing a crucial role in its ability to **improve and adapt** over time.

1. **Learning Element** → Responsible for **making improvements** based on feedback.
2. **Performance Element** → Selects **external actions** based on the current knowledge.
3. **Critic** → Provides **feedback** on performance, guiding the **Learning Element** to modify behavior for **better future decisions**.
4. **Problem Generator** → Suggests **new actions** that lead to **informative experiences** and exploration.
    - Similar to **scientists conducting experiments** to **discover new knowledge**.

Real-World Analogy: Learning in School

- A **test** serves as the **Critic**, evaluating **what can be improved**.
- The **teacher** acts as the **Learning Element**, identifying **mistakes** and **instructing improvements**.
- The **student** represents the **Performance Element**, applying **feedback** in future tests.
- The **Problem Generator** is like a **teacher suggesting new experiments**, such as placing a **mass on a spring**, which leads to **new insights**.

This structured approach allows **AI systems to refine their decisions** and **enhance learning efficiency** over time.

Learning in Intelligent Agents

Learning in **intelligent agents** involves **modifying** each **component** of the agent to **align** with **feedback information**, thereby **enhancing performance**.

A **human** is an example of a **learning agent**.

- **Example**: Learning to **ride a bicycle**—humans are not born with this skill, but they can **acquire** it through **experience** and **practice**.

Applications of Learning Agents in AI

1. **Search Engines (Google)** → AI learns from **user queries and interactions** to **refine search results**.
2. **Computer Vision** → Automates tasks performed by the **human visual system**, enabling applications like **image recognition**.
3. **Self-Driving Cars** → AI **learns** to interpret **sensor data**, predict **traffic behavior**, and navigate **autonomously**.
4. **Recognition of Gestures** → A type of **perceptual computing** where computers **capture and interpret** human gestures as **commands**.

These applications demonstrate how **learning agents** continually **improve** and **adapt** to perform tasks more efficiently.

Comparison of Different Types of Learning Agents

Feature	Reflex Agent	Model-Based Agent	Goal-Based Agent	Utility-Based Agent	Learning Agent
Definition	Acts based on condition-action rules (if-then)	Uses internal models to maintain state	Chooses actions to achieve a goal	Chooses actions based on expected utility	Improves performan over time l learning fr experienc
Memory Usage	No memory, purely reactive	Maintains internal state	Maintains state and goals	Maintains state, goals, and utility values	Learns an adapts memory dynamica
Adaptability	Not adaptable, fixed rules	Limited adaptability based on state	More adaptable with goal selection	High adaptability based on utility calculations	Highly adaptable learns fro data
Complexity	Low (simple rules)	Moderate (state tracking)	Higher (goal setting)	High (utility evaluation)	Highest (learning generaliza
Decision Making	Based on current percepts only	Uses past information to make decisions	Evaluates actions based on goals	Evaluates best action based on utility	Adjusts decisions based on learning
Handling New Situations	Poor (not flexible)	Moderate (can update state)	Good (goal-driven)	Better (optimizes outcomes)	Best (continuou improves)
Example	Thermostat	Self-driving car tracking past routes	Chess-playing AI aiming for checkmate	AI recommending best stock investment	Deep lear based spe recognitio
Computational Cost	Low	Moderate	High	Higher	Very High
Goal Orientation	No explicit goals	Uses state to	Explicit goal-	Uses optimization	Learns to optimize

Feature	Reflex Agent	Model-Based Agent	Goal-Based Agent	Utility-Based Agent	Learning Agent
		approximate goals	seeking behavior	for best outcome	based on experienc
Learning Capability	None	Limited	Limited	Indirect (through optimization)	Fully capa of learnin

Let me know if you need modifications! 🚀

PEAS Description of an Online Shopping Agent

The **PEAS (Performance, Environment, Actuators, Sensors)** framework defines an **intelligent agent's** components in a given task.

PEAS for Shopping for Data Warehousing Books Online

Performance Measures

- **Price of the book** → Ensures affordability and cost-effectiveness.
- **Author of the book** → Determines credibility and expertise.
- **Quality of the book** → Evaluates content relevance and reliability.
- **Book reviews on Google** → Assesses user feedback and ratings.
- **Obtain interested/desired books** → Ensures the agent finds the correct book.
- **Cost minimization** → Finds the best price across different vendors.

Environment

- **Internet websites** → Platforms where books are available.
- **Web pages of a particular website** → Individual book listings and details.
- **Vendors/Sellers** → Online marketplaces like Amazon, Flipkart, etc.
- **Shippers** → Delivery service providers handling book shipments.

Actuators

- **Filling in forms** → Enters search queries, payment, and delivery details.
- **Display to the user** → Presents search results, recommendations, and prices.
- **Follow URL** → Navigates through links to access book listings and purchase pages.

Sensors

- **Keyboard entry** → Captures user inputs like search queries and filters.
- **Browser used to find web pages** → Retrieves and loads website data.
- **HTML** → Extracts book details from structured web pages.

The **PEAS model** helps define how an **online shopping agent** operates, processes data, and interacts with its environment to **efficiently** find, compare, and purchase books.

Task Environment

The **task environment** describes the characteristics of the world in which an **AI agent** operates.

1. Observable (Fully or Partially)

- **Partially Observable** → The agent **cannot** determine the **complete state** of the environment at all times.
- **Example** → The shopping agent cannot view **all types of books** on a single webpage.
  - If the user wants **high-rated books**, the agent must **navigate** to a new page or **apply filters**.
  - Since the agent **does not have complete visibility**, the environment is **partially observable**.

2. Deterministic or Non-deterministic

- **Deterministic** → The **current state** and **actions** fully determine the **next state**.
- **Example** → If the shopping agent selects a book:
  - The next steps → **Payment, delivery address, order confirmation**.
  - Since each action leads to a **fixed next state**, the environment is **deterministic**.

3. Episodic or Sequential

- **Sequential** → Actions in the **current state affect future states**.
- **Example** → If the agent **rejects a book**, it will **not appear** in future recommendations.
  - The **webpage updates dynamically** based on user preferences.
  - Since actions in one state **influence** future states, the environment is **sequential**.

4. Static or Dynamic

- **Static** → The environment **does not change** over time.
- **Example** → The **book listings** on the website **remain unchanged** unless refreshed manually.

- Unlike a **car-driving environment** (which is dynamic), the **shopping website stays static**.

5. Discrete or Continuous

- **Discrete** → The environment consists of a **finite number of states**.
- **Example** → The shopping agent operates in a structured sequence:
  1. **View book details**
  2. **Check the price**
  3. **Fill in the purchase form**
  4. **Place an order and make payment**
  - Since these actions are **countable and distinct**, the environment is **discrete**.

6. Single-Agent or Multi-Agent

- **Single-Agent** → Only **one agent** interacts with the environment.
- **Example** → The shopping agent **alone** interacts with the website.
  - In contrast, a **chess game** is a **multi-agent** environment (two players).
  - Since no **other AI agents** are present, the shopping task is a **single-agent system**.

PEAS for Bidding on an Item at an Auction

The **PEAS (Performance Measure, Environment, Actuators, Sensors)** framework describes an AI agent's interaction with its environment.

1. Performance Measure

The goal of the agent is to **maximize bidding efficiency** while minimizing unnecessary costs. The agent's performance is measured by:

- **Winning the auction** at the **lowest possible bid**.
- **Avoiding overbidding** beyond the predefined budget.
- **Responding quickly** to competing bids.
- **Following auction rules** and not violating bidding constraints.

2. Environment

The **auction setting** where the bidding takes place:

- **Types of Auctions** → Online auctions (e.g., eBay), live in-person auctions, or silent auctions.

- **Competing bidders** (human or automated AI agents).
- **Auctioneer** managing the bidding process.
- **Time limits** for placing bids.
- **Auction rules** (e.g., minimum bid increments, reserve prices, auto-bidding).

3. Actuators (Actions the agent can perform)

- **Placing a bid** at the right moment.
- **Adjusting bid values** based on competition.
- **Withdrawing from bidding** when exceeding budget constraints.
- **Using auto-bidding strategies** to outbid competitors efficiently.

4. Sensors (Data sources for decision-making)

- **Current highest bid** (to determine the next bid amount).
- **Number of competing bidders** (to assess competition level).
- **Time remaining** in the auction.
- **Historical bidding patterns** of competitors.
- **Auction rules and constraints** (e.g., bid increments, max bid limit).

The AI **bidding agent** operates by **analyzing the environment**, sensing the competition, and strategically placing bids to **win efficiently** while staying within budget.

PEAS for Bidding on an Item at an Auction

1. Performance Measures

The agent's success is evaluated based on:

- **Cost of the item** (minimizing expenditure while securing the item).
- **Quality of the item** (ensuring the item meets expected standards).
- **Value of the item** (assessing worth relative to price).
- **Necessity of the item** (determining whether the purchase is essential).

2. Environment

The auction setting includes:

- **Auctioneer** (conducting the bidding process).
- **Bidders** (competing for the item).
- **Items for bidding** (various products available for purchase).

3. **Actuators** (Means to perform actions)

- **Speakers** (announcing bids and item details).
- **Microphones** (for verbal bidding).
- **Display items** (showing the current highest bid).
- **Budget control** (managing available funds).

4. **Sensors** (Means to perceive the environment)

- **Camera** (monitoring the auction and bidders).
- **Price monitor** (tracking bid updates).
- **Eyes** (observing auction events and competitors).
- **Ears** (listening to auctioneer announcements and competing bids).

Task Environment for Bidding on an Item at an Auction

1. Observability (Fully/Partially Observable)

- **Partially Observable:** The agent cannot determine the complete state of the environment at all times.
- The auctioneer does not have full knowledge of bidder strategies or intentions.
- Any environment involving human participants is typically **partially observable**.

2. Agents (Single/Multi-Agent)

- **Single-Agent:** The auction agent operates independently.
- Other human bidders exist, but they are not part of the agent's decision-making system.
- The agent only processes perceptual data from multiple bidders.

3. Deterministic vs. Stochastic

- **Stochastic:** The outcome is uncertain and cannot be determined solely by the agent’s current state.
- Bidding introduces randomness as different bidders make unpredictable decisions.

4. Episodic vs. Sequential

- **Sequential:** Each action affects future outcomes.
- If one bidder places a bid of **X**, the next bidder must bid **higher** than **X** or drop out.
- The episodes are **dependent**, making it a **sequential task**.

5. Static vs. Dynamic

- **Dynamic:** The environment is constantly changing.
- Prices fluctuate based on competing bids, making it an ever-evolving scenario.
- A **static environment** (e.g., crossword puzzles) does not change over time, but auctions do.

6. Discrete vs. Continuous

- **Continuous:** The number of possible states is **unbounded**.
- Bidders can keep placing new values indefinitely, leading to an unpredictable number of states.
- Unlike **discrete environments** (e.g., chess, which has a limited number of moves), auctions can have **unlimited price variations**.

Made By Yashank