

ATME COLLEGE OF ENGINEERING

13th KM Stone, Bannur Road, Mysore - 560 028



A T M E
College of Engineering

DEPARTMENT OF COMPUTER SCIENCE AND DESIGN



LECTURE NOTES

COURSE: Artificial Intelligence

COURSE CODE: BCS515B

SEMESTER: V

COURSE COORDINATOR: Mrs. DIVYA N

(ACADEMIC YEAR 2025-26)

INSTITUTIONAL VISION AND MISSION

Vision

Development of academically excellent, culturally vibrant, socially responsible and globally competent human resources.

Mission

- To keep pace with advancements in knowledge and make the students competitive and capable at the global level.
- To create an environment for the students to acquire the right physical, intellectual, emotional and moral foundations and shine as torch bearers of tomorrow's society.
- To strive to attain ever-higher benchmarks of educational excellence.

DEPARTMENT VISION AND MISSION

Vision

To be a global leader in Computer Science and Design Engineering striving for design excellence, cultural awareness, a profound commitment to environmental stewardship, and societal responsibility.

Mission

M1: To establish a technology-enabled experiential learning environment, prioritizing and cultivating problem-solving and design thinking skills among students.

M2: To foster collaboration with industries, research and development organizations, jointly addressing socially relevant challenges in Computer Science with a core emphasis on design.

Program Educational Objective (PEO's)

- Demonstrate the proficiency and adaptability needed to seamlessly integrate with the latest advancements and changes in Computer Science and Design.
- Our graduates will possess the adaptability to explore various career paths, such as Research & Development, Manufacturing, and Entrepreneurship with independent design firms, Design Services, and Consultancies.
- Our students will be well-prepared to excel in collaborative, multi-disciplinary environments, showcasing their versatility and skills.

Program Specific Outcomes (PSO's)

- To develop stand-alone, embedded, and web-based solutions with easy-to-operate interfaces using software engineering practices and contemporary programming languages.
- Design and develop computer-based systems in various areas of Multimedia, Graphics data visualization and computer vision.

MODULE - 1

Introduction to Artificial Intelligence:

AI is one of the newest fields in science and engineering. Work started in earnest soon after World War II, and the name itself was coined in 1956. AI currently encompasses a huge variety of subfields, ranging from the general (learning and perception) to the specific, such as playing chess, proving mathematical theorems, writing poetry, driving a car on a crowded street, and diagnosing diseases. AI is relevant to any intellectual task; it is truly a universal field.

1.1 WHAT IS AI?

In Figure 1.1 we see eight definitions of AI, laid out along two dimensions. The definitions on top are concerned with thought processes & reasoning, whereas the ones on the bottom address behavior. The definitions on the left measure success in terms of fidelity to human performance, whereas the ones on the right measure against an ideal performance measure, called rationality. A system is rational if it does the —right thing, given what it knows. Historically, all four

Thinking Humanly “The exciting new effort to make computers think ... <i>machines with minds</i> , in the full and literal sense.” (Haugeland, 1985) “[The automation of] activities that we associate with human thinking, activities such as decision-making, problem solving, learning ...” (Bellman, 1978)	Thinking Rationally “The study of mental faculties through the use of computational models.” (Charniak and McDermott, 1985) “The study of the computations that make it possible to perceive, reason, and act.” (Winston, 1992)
Acting Humanly “The art of creating machines that perform functions that require intelligence when performed by people.” (Kurzweil, 1990) “The study of how to make computers do things at which, at the moment, people are better.” (Rich and Knight, 1991)	Acting Rationally “Computational Intelligence is the study of the design of intelligent agents.” (Poole <i>et al.</i> , 1998) “AI ... is concerned with intelligent behavior in artifacts.” (Nilsson, 1998)

Figure 1.1 Some definitions of artificial intelligence, organized into four categories.

approaches to AI have been followed, each by different people with different methods. A human-centered approach must be in part an empirical science, involving observations and hypotheses about human behavior. A rationalist approach involves a combination of mathematics and engineering.

1.1.1 Acting humanly: The Turing Test approach TURING TEST

The Turing Test, proposed by Alan Turing (1950), was designed to provide a satisfactory operational definition of intelligence. A computer passes the test if a human interrogator, after posing some written questions, cannot tell whether the written responses come from a person or from a computer. The computer would need to possess the following capabilities:

- Natural language processing to enable it to communicate successfully in English;
- Knowledge representation to store what it knows or hears;
- Automated reasoning to use the stored information to answer questions and to draw conclusions;
- Machine learning to adapt one circumstances and to detect and extrapolate patterns.

Turing's test deliberately avoided direct physical interaction between the interrogator and the computer, because physical simulation of a person is unnecessary for intelligence. However, the so-called total Turing Test includes a video signal so that interrogator can test the subject's perceptual abilities, as well as the opportunity for the interrogator to pass physical objects—through the hatch.¶ To pass the total Turing Test, the computer will need

- Computer vision to perceive objects, and
- Robotics to manipulate objects and move about.

1.1.2 Thinking humanly: The cognitive modeling approach

If we are going to say that a given program thinks like a human, we must have some way of determining how humans think. We need to get inside the actual workings of human minds. There are three ways to do this: through introspection—trying to catch our own thoughts as they go by; through psychological experiments—observing a person in action; and through brain imaging—observing the brain in action . Once we have a sufficiently precise theory of the mind, it becomes possible to express the theory as a computer program. If the program's input–output behavior matches corresponding human behavior, that is evidence that some of the program's mechanisms could also be operating in humans.

1.1.3 Thinking rationally: The “laws of thought” approach

The Greek philosopher Aristotle was one of the first to attempt to codify —right thinking,¶ that is, certain reasoning processes. His syllogisms provided patterns for argument structures that always yielded correct conclusions when given correct premises. Logicians in the 19th century developed a precise notation for statements about all kinds of objects in the world

and the relations among them. The so-called logic is tradition within artificial intelligence hopes to build on such programs to create intelligent systems. There are two main obstacles to this approach. First, it is not easy to take informal knowledge and state it in the formal terms required by logical notation, particularly when the knowledge is less than 100% certain. Second, there is a big difference between solving a problem—in principle and solving it in practice.

1.1.4 Acting rationally: The rational agent approach

An agent is just something that acts (agent comes from the Latin *agere*, to do). Of course, all computer programs do something, but computer agents are expected to do more: operate autonomously, perceive their environment, persist over a prolonged time period, adapt to change, and create and pursue goals. A rational agent is one that acts so as to achieve the best outcome or, when there is uncertainty, the best expected outcome. In the—laws of thought approach to AI, the emphasis was on correct inferences. Making correct inferences is sometimes part of being a rational agent, because one way to act rationally is to reason logically to the conclusion that a given action will achieve one's goals and then to act on that conclusion.

The rational agent approach has advantages over the other approaches. First, it is more general than the—laws of thought approach because correct inference is just one of several possible mechanisms for achieving rationality. Second, it is more amenable to scientific development than are approaches based on human behavior or human thought. The standard of rationality is mathematically well defined and completely general and can be—unpacked to generate agent designs that provably achieve it.

1.4 The State of The Art

What can AI do today? A concise answer is difficult because there are so many activities in so many subfields. Here we sample a few applications:

Robotic vehicles:

A driverless robotic car named STANLEY sped through the rough terrain of the Mojave desert at 22 mph, finishing the 132-mile course first to win the 2005 DARPA Grand Challenge. STANLEY is a Volkswagen Touareg outfitted with cameras, radar, and laser rangefinders to sense the environment and onboard software to command the steering, braking, and

acceleration (Thrun, 2006). The following year CMU's BOSS won the Urban Challenge, safely driving in traffic through the streets of a closed Air Force base, obeying traffic rules and avoiding pedestrians and other vehicles.

Speech recognition:

A traveler calling United Airlines to book a flight can have the entire conversation guided by an automated speech recognition and dialog management system.

Autonomous planning and scheduling:

A hundred million miles from Earth, NASA's Remote Agent program became the first on-board autonomous planning program to control the scheduling of operations for a spacecraft (Jonsson et al., 2000). REMOTE AGENT generated plans from high-level goals specified from the ground and monitored the execution of those plans—detecting, diagnosing, and recovering from problems as they occurred. Successor program MAPGEN (Al-Chang et al., 2004) plans the daily operations for NASA's Mars Exploration Rovers, and MEXAR2 (Cesta et al., 2007) did mission planning—both logistics and science planning—for the European Space Agency's Mars Express mission in 2008

Game playing:

IBM's DEEP BLUE became the first computer program to defeat the world champion in a chess match when it bested Garry Kasparov by a score of 3.5 to 2.5 in an exhibition match (Goodman and Keene, 1997). Kasparov said that he felt a —new kind of intelligence across the board from him. Newsweek magazine described the match as —The brain's last stand. The value of IBM's stock increased by \$18 billion. Human champions studied Kasparov's loss and were able to draw a few matches in subsequent years, but the most recent human-computer matches have been won convincingly by the computer.

Spam fighting: Each day, learning algorithms classify over a billion messages as spam, saving the recipient from having to waste time deleting what, for many users, could comprise 80% or 90% of all messages, if not classified away by algorithms. Because the spammers are continually updating their tactics, it is difficult for a static programmed approach to keep up, and learning algorithms work best (Sahami et al., 1998; Goodman and Heckerman, 2004).

Logistics planning:

During the Persian Gulf crisis of 1991, U.S. forces deployed a Dynamic Analysis and Replanning Tool, DART (Cross and Walker, 1994), to do automated logistics planning and scheduling for transportation. This involved up to 50,000 vehicles, cargo, and people at a time, and had to account for starting points, destinations, routes, and conflict resolution among all parameters. The AI planning techniques generated in hours a plan that would have taken weeks

with older methods. The Defense Advanced Research Project Agency (DARPA) stated that this single application more than paid back DARPA's 30-year investment in AI.

Robotics: The iRobot Corporation has sold over two million Roomba robotic vacuum cleaners for home use. The company also deploys the more rugged PackBot to Iraq and Afghanistan, where it is used to handle hazardous materials, clear explosives, and identify the location of snipers.

Machine Translation: A computer program automatically translates from Arabic to English, allowing an English speaker to see the headline —Ardogan Confirms That Turkey Would Not Accept Any Pressure, Urging Them to Recognize Cyprus.¶ The program uses a statistical model built from examples of Arabic-to-English translations and from examples of English text totaling two trillion words (Brants et al., 2007). None of the computer scientists on the team speak Arabic, but they do understand statistics and machine learning algorithms.

These are just a few examples of artificial intelligence systems that exist today. Not magic or science fiction—but rather science, engineering, and mathematics.

Chapter-02: - Problem-solving agents

Agents and Environment:

An agent is anything that can be viewed as perceiving its environment through sensors and SENSOR acting upon that environment through actuators. This simple idea is illustrated in Figure 2.1. ACTUATOR A human agent has eyes, ears, and other organs for sensors and hands, legs, vocal tract, and so on for actuators.

We use the term percept to refer to the agent's perceptual inputs at any given instant. An PERCEPT SEQUENCE agent's percept sequence is the complete history of everything the agent has ever perceived.

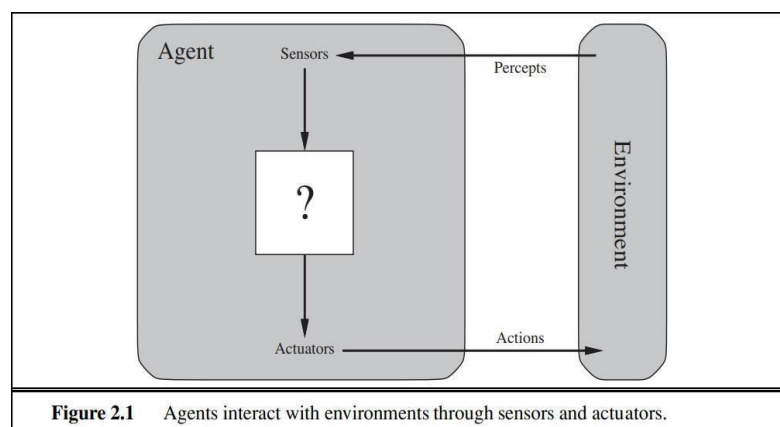


Figure 2.1 Agents interact with environments through sensors and actuators.

Mathematically speaking, we say that an agent's behavior is AGENT FUNCTION described by the agent function that maps any given percept sequence to an action.

We can imagine tabulating the agent function that describes any given agent;

Given an agent to experiment with, we can, in principle, construct this table by trying out all possible percept sequences and recording which actions the agent does in response. The table is, of course, an external characterization of the agent. Internally, the agent function for an artificial agent will be implemented by an AGENT PROGRAM agent program.

It is important to keep these two ideas distinct. The agent function is an abstract mathematical description; the agent program is a concrete implementation, running within some physical system.

The Vacuum-Cleaner World:

- This world is so simple that we can describe everything that happens; it's also a made-up world, so we can invent many variations. This particular world has just two locations: squares A and B. The vacuum agent perceives which square it is in and whether there is dirt in the square. It can choose to move left, move right, suck up the dirt, or do nothing. One very simple agent function is the following: if the current square is dirty, then suck; otherwise, move to the other square.

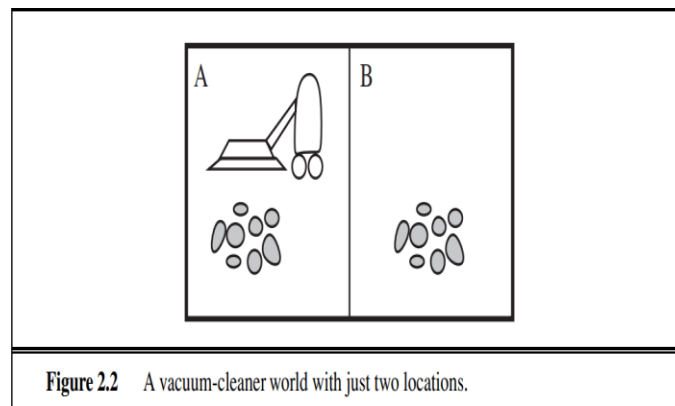


Figure 2.2 A vacuum-cleaner world with just two locations.

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.

Program implements the agent function tabulated in Fig. 2.3

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***

2.2 Good Behavior: The Concept of Rationality

Rationality in AI agents involve performing the "right" actions to achieve desirable outcomes, which are evaluated through a performance measure. This measure assesses the desirability of the resulting states in the environment, rather than the agent's own perception of success. A key challenge is designing performance measures that truly reflect the goals of the environment, as poor design can lead to unintended behaviors. For example, a vacuum-cleaner agent might focus on cleaning up dirt repeatedly rather than maintaining a clean floor. Additionally, complex philosophical questions arise, such as whether consistent mediocre performance is preferable to alternating highs and lows. Ultimately, rationality in agents depends on aligning their actions with well-defined performance measures that ensure the desired outcomes are achieved in the environment.

2.2.1 Rationality

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

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

This leads to a definition of a rational agent: For each possible percept sequence, a rational agent should select an action that is expected to maximize its performance measure, given the evidence provided by the percept sequence and whatever built-in knowledge the agent has.

2.2.2 Omniscience, learning, and autonomy

It's important to distinguish between rationality and omniscience. An omniscient agent would know the exact outcomes of its actions and could act accordingly, but in reality, omniscience is impossible. For example, if I decide to cross a street after seeing a friend and, unexpectedly, a cargo door falls from an airplane and hits me, this does not mean my decision to cross the street was irrational. Rationality involves making decisions that maximize expected performance based on available information, not achieving perfect outcomes. Expecting an

agent to always make the best possible decision after the fact is unrealistic, as it would require knowledge of future events—something only a time machine or crystal ball could provide. Rationality is about doing the best one can with the information at hand, not achieving flawless outcomes.

Our definition of rationality does not require omniscience because rationality depends only on the percept sequence received by the agent so far, not on future knowledge. However, rationality also means avoiding actions that are underinformed or risky. For example, if an agent doesn't look both ways before crossing a busy road, it lacks crucial information, and crossing would be irrational. A rational agent should engage in information gathering, such as looking before crossing, to maximize expected performance. Information gathering and exploration are key aspects of rationality, as they help the agent improve decision-making by acquiring necessary data.

Additionally, rational agents must not only gather information but also learn from their experiences. While an agent may start with some prior knowledge, it should adapt and update its understanding as it interacts with the environment. This learning process helps the agent become more autonomous, reducing reliance on initial assumptions and making it more effective in diverse environments. For instance, a vacuum-cleaning agent that learns where dirt typically appears will perform better than one that does not.

Autonomy is essential for rationality. An agent that depends too much on its designer's prior knowledge without learning from its own percepts lacks autonomy. Over time, a rational agent should be able to act independently of its initial programming, adjusting its behavior based on experience. This combination of initial knowledge and continuous learning enables the design of a single rational agent capable of succeeding in a wide range of environments.

2.3 The nature of environment

Now that we understand what rationality means, the next step in creating rational agents is to think about **task environments**. A task environment is basically the "problem" that the agent is designed to solve. Before we can build a rational agent, we need to clearly define the task environment, including what the agent needs to achieve, what information it receives, and what actions it can take. Different environments have different needs, so the way we design the agent depends on the specific environment it will operate in. In the next sections, we'll look at examples of task environments and see how they influence the design of the agent.

2.3.1 Specifying the task environment:

When discussing the rationality of the simple vacuum-cleaner agent, we had to define four key elements: the **Performance measure** (how success is judged), the **Environment** (where the agent operates), the **Actuators** (the agent's actions), and the **Sensors** (how the agent perceives its surroundings). These elements are grouped under the heading of the **task environment**, and we use the acronym **PEAS** (Performance, Environment, Actuators, Sensors) to remember them. When designing any agent, the first step is to fully specify its task environment using this PEAS framework.

For example, consider the more complex problem of designing an automated taxi driver. Unlike the simple vacuum-cleaner scenario, the task of driving a taxi is extremely open-ended, with countless possible situations that the agent might encounter. While fully automated taxis are still beyond current technology, this example highlights the importance of carefully defining the task environment to ensure the agent is designed effectively. By specifying the performance measures, environment, actuators, and sensors, we can better understand the challenges the agent will face and how it should be designed to handle them.

Figure 2.4 summarizes the PEAS description for the taxi's task environment.

Agent Type	Performance Measure	Environment	Actuators	Sensors
Taxi driver	Safe, fast, legal, comfortable trip, maximize profits	Roads, other traffic, pedestrians, customers	Steering, accelerator, brake, signal, horn, display	Cameras, sonar, speedometer, GPS, odometer, accelerometer, engine sensors, keyboard
Figure 2.4 PEAS description of the task environment for an automated taxi.				

2.3.2 Properties of task environments

The range of task environments in AI is incredibly broad, but we can categorize them using a small set of key **dimensions**. These dimensions help determine the best way to design the agent and choose the appropriate techniques for implementation. By understanding these dimensions, we can better match the agent's design to the specific challenges it will face in different environments. First, we'll list these dimensions, and then we'll analyze several task environments to see how these ideas apply in practice. While the definitions provided here are informal, later chapters will offer more detailed explanations and examples of each type of

environment.

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.				

2.4 The structure of agents:

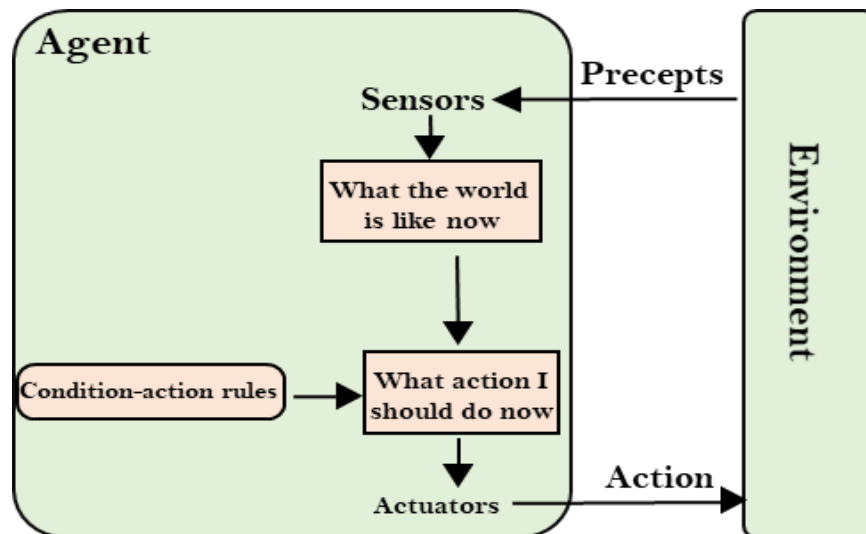
- Agents can be grouped into five classes based on their degree of perceived intelligence and capability.
- All these agents can improve their performance and generate better action over the time. These are given below:

1) Simple Reflex Agent

The Simple reflex agents are the simplest agents. These agents take decisions on the basis of the current precepts and ignore the rest of the percept history.

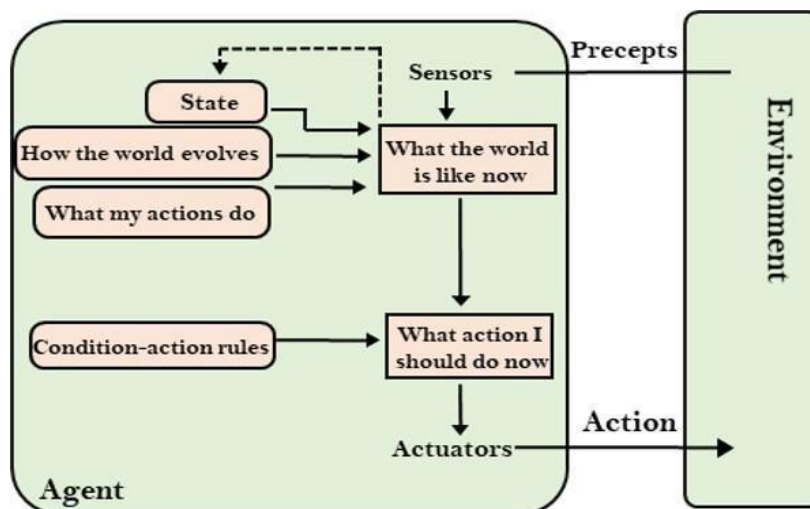
These agents only succeed in the fully observable environment.

The Simple reflex agent does not consider any part of precepts history during their decision and action process.



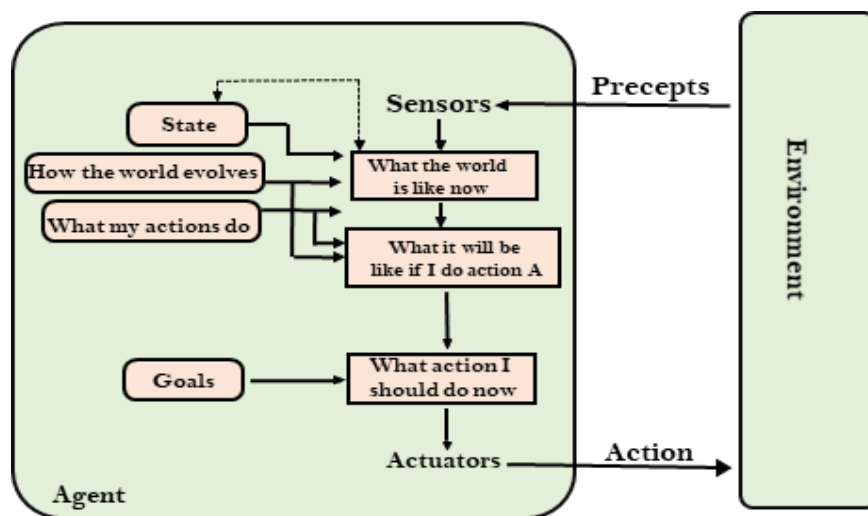
- **Model-based reflex agent**

- The Model-based agent can work in a partially observable environment, and track the situation.
- A model-based agent has two important factors:
Model: It is knowledge about "how things happen in the world," so it is called a Model-based agent.
- Internal State: It is a representation of the current state based on percept history.
- These agents have the model, "which is knowledge of the world" and based on the model they perform actions.
- Updating the agent state requires information about:
 - How the world evolves
 - How the agent's action affects the world.



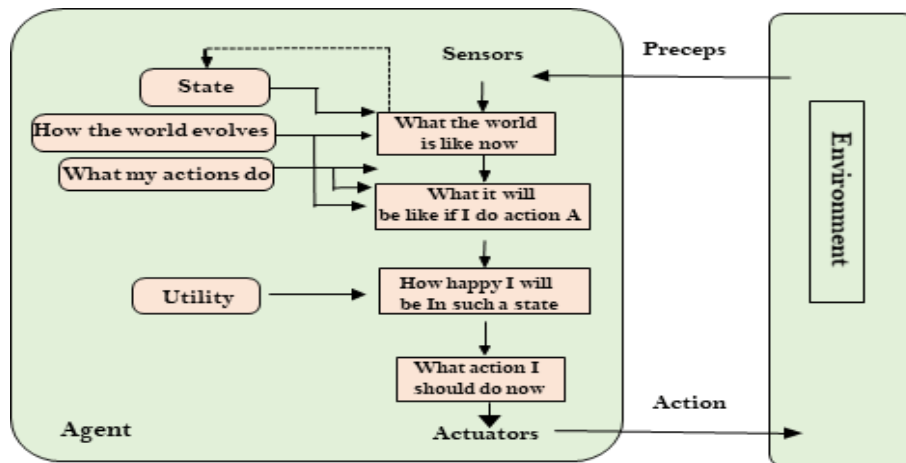
2. Goal-based agents

- The knowledge of the current state environment is not always sufficient to decide for an agent to what to do.
- The agent needs to know its goal which describes desirable situations.
- Goal-based agents expand the capabilities of the model-based agent by having the "goal" information.
- They choose an action, so that they can achieve the goal.
- These agents may have to consider a long sequence of possible actions before deciding whether the goal is achieved or not. Such considerations of different scenario are called searching and planning, which makes an agent proactive.



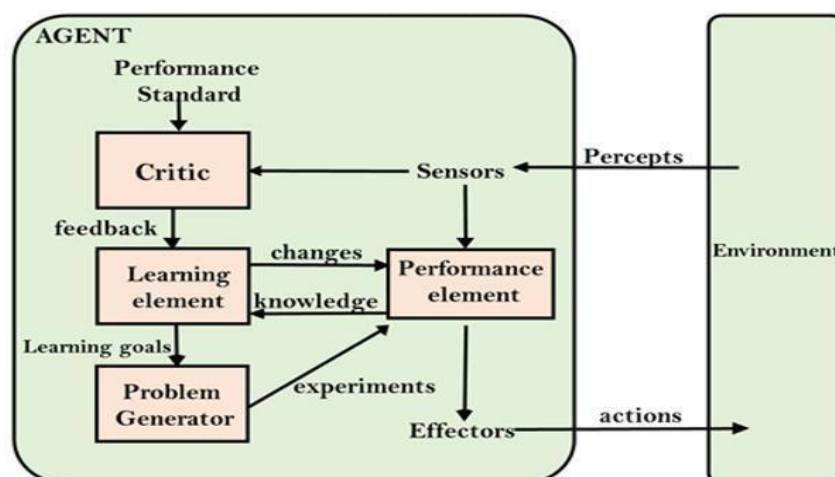
3. Utility-based agent

- These agents are similar to the goal-based agent but provide an extra component of utility measurement which makes them different by providing a measure of success at a given state.
- Utility-based agent act based not only goals but also the best way to achieve the goal.
- The Utility-based agent is useful when there are multiple possible alternatives, and an agent has to choose in order to perform the best action.
- The utility function maps each state to a real number to check how efficiently each action achieves the goals.



4. Learning agent

- A learning agent in AI is the type of agent which can learn from its past experiences, or it has learning capabilities.
- It starts to act with basic knowledge and then able to act and adapt automatically through learning.
- A learning agent has mainly four conceptual components, which are:
 - **Learning element:** It is responsible for making improvements by learning from environment
 - **Critic:** Learning element takes feedback from critic which describes that how well the agent is doing with respect to a fixed performance standard.
 - **Performance element:** It is responsible for selecting external action
 - **Problem generator:** This component is responsible for suggesting actions that will lead to new and informative experiences.



How the components of agent programs work:

We've discussed agent programs at a high level, focusing on their components that answer questions like "What is the world like now?" and "What action should I take?" Now, the next step for AI students is to understand how these components actually work. This is a complex topic that requires much deeper exploration, but we can start by highlighting some basic distinctions in how these components represent the environment.

These representations can be categorized along a spectrum of complexity and expressive power: **atomic**, **factored**, and **structured**. This classification helps us understand how an agent models its environment and decides on actions. For example, consider the component that handles "What my actions do"—this part of the agent program represents the changes in the environment that result from the agent's actions. By understanding these different levels of representation, we can begin to grasp the various ways in which an agent can interpret and interact with its environment. Figure 2.16 provides schematic depictions of how those transitions might be represented.

