PDS 4505 Artificial Intelligence

Dr.S.Jothilakshmi
Asst.Professor
Dept of Data Science

Introduction

- What is Artificial Intelligence?
  - Systems that think like humans
  - Systems that act like humans
  - Systems that act or think rationally (logically, correctly)

WHAT is AI?

- "The exciting new effort to make computers think . . . machines with minds((Haugeland, 1985)
- The automation of] activities that we associate with human thinking, activities such as decision-making, problem solving, learning
- The study of mental abilities through the use of computational models“
- The study of the computations that make it possible to perceive, reason, and act"
- The art of creating machines that perform functions that require intelligence when performed by people
- The study of how to make computers do things at which, at the moment, people are better
- A field of study that seeks to explain and emulate intelligent behavior in terms of computational processes
- The branch of computer science that is concerned with the automation of intelligent behavior

- John McCarthy coined the phrase Artificial Intelligence (AI) in 1956
- It is the science and engineering of making intelligent machines, especially intelligent computer programs.
- It is related to the similar task of using computers to understand human or *other* intelligence
- AI is a collection of hard problems which can be solved by humans and other living things, but for which we don't have good algorithms for solving.
  - e. g., understanding spoken natural language, medical diagnosis, chess playing, proving math theories and many more.
- Intelligence is the computational part of the ability to achieve goals in the world.

What is Intelligence?

The Turing Test

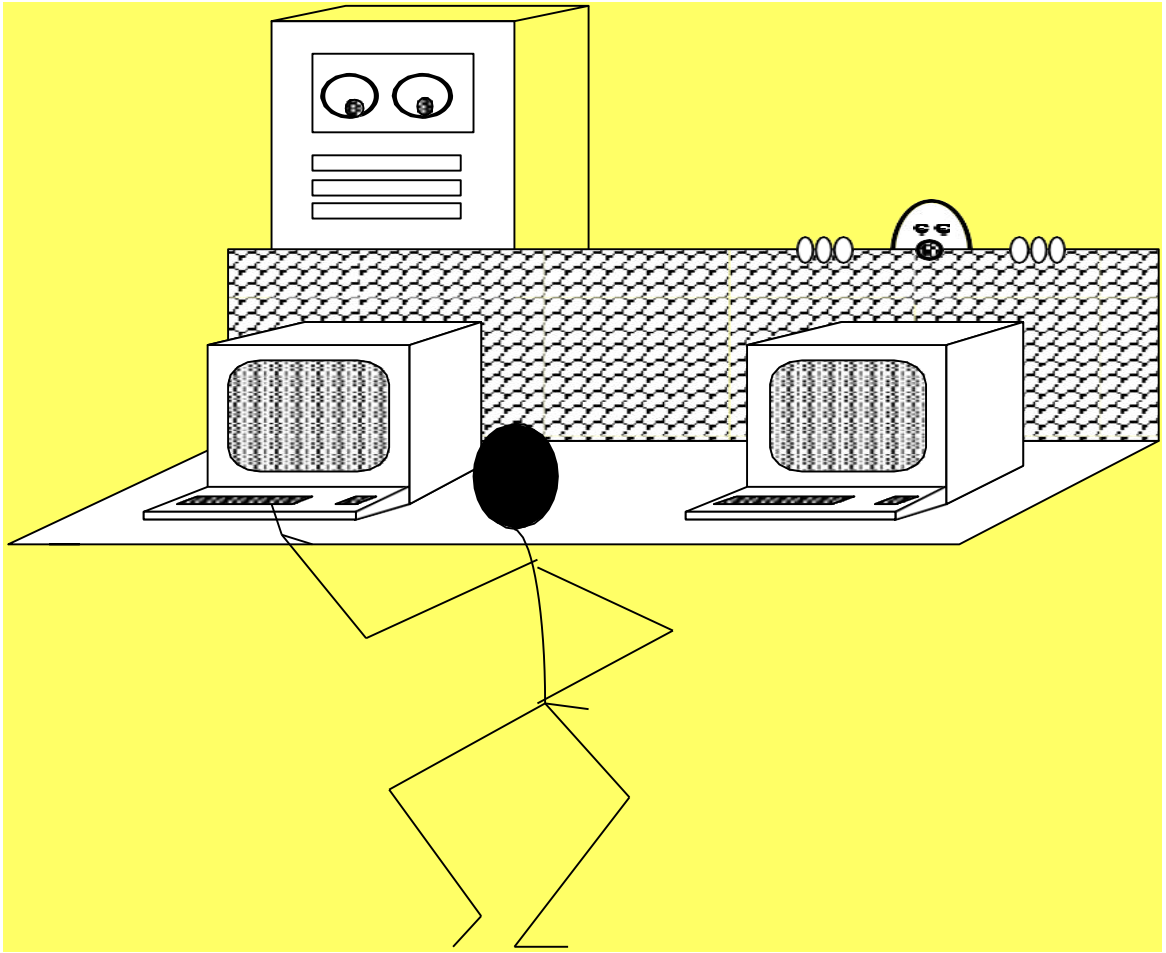


Fig.1 Turing Machine

A machine can be described as a thinking machine if it passes the Turing Test. i.e. If a human agent is engaged in two isolated dialogues (connected by teletype say); one with a computer, and the other with another human and the human agent cannot reliably identify which dialogue is with the computer.

- Some definitions of AI. They are organized into four categories:
- Systems that think like humans.
- Systems that act like humans.
- Systems that think rationally.
- Systems that act rationally.

Acting humanly: The Turing Test approach

The total Turing Test includes a video signal so that the interrogator can test the subject's perceptual abilities, "

To pass the total Turing Test, the computer will need

COMPUTER VISION robotics to move them about.

- natural language processing to enable it to communicate successfully in English (or some other human language);
- knowledge representation to store information provided before or during the interrogation;
- automated reasoning to use the stored information to answer questions and to draw new conclusions;
- machine learning to adapt to new circumstances and to detect and extrapolate patterns

Thinking humanly: The cognitive modelling approach

- program thinks like a human
- The interdisciplinary field of cognitive science brings together computer models from AI and experimental techniques from
- psychology to try to construct precise and testable theories of the workings of the human mind

Thinking rationally: The laws of thought approach

- These laws of thought were supposed to govern the operation of the mind, and initiated the field of logic.
- 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 being able to solve a problem "in principle"

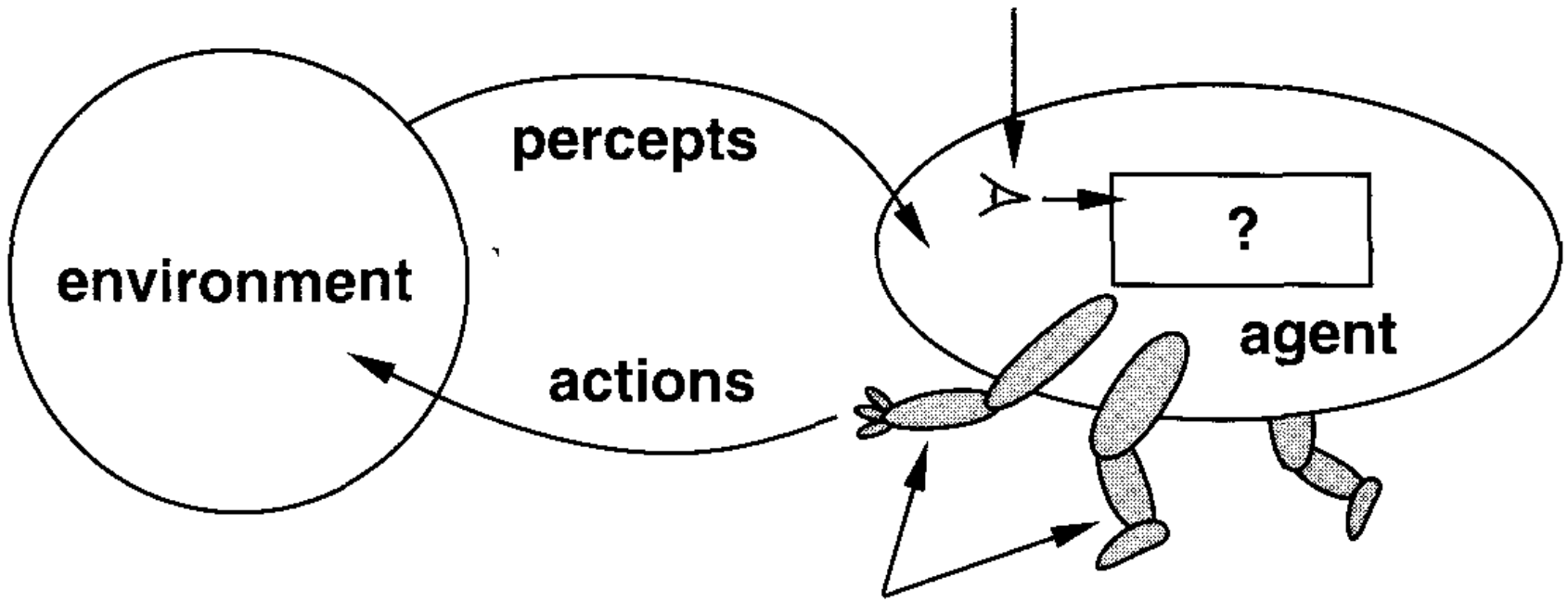
Acting rationally: The rational agent approach

- Acting rationally means acting so as to achieve one's goals, given one's beliefs. An agent is just something that perceives and acts.
- In this approach, AI is viewed as the study and construction of rational agents

INTELLIGENT AGENTS

- **An agent is anything that can be viewed as perceiving its environment through sensors and acting** upon that environment through **effectors**
- design agents that do a good job of acting on their environment.
- A human agent has eyes, ears, and other organs for sensors, and hands, legs, mouth, and other body parts for effectors.
- A robotic agent substitutes cameras and infrared range finders for the sensors and various motors for the effectors

How AGENTS SHOULD ACT



The rational agents depends on four things:

- • The performance measure that defines degree of success.
- • Everything that the agent has perceived so far. We will call this complete perceptual history
- the **percept sequence.**
- • What the agent knows about the environment.
- • The actions that the agent can perform.

The ideal mapping from percept sequences to actions

- we can describe any particular agent by making a table of the action it takes in response to each possible percept sequence.
- Such a list is called a **mapping from percept sequences to actions**

- if mappings describe agents, then **ideal mappings describe ideal agents.**
- *Specifying which action an agent ought to take in response to any given percept sequence provides a design for an ideal agent.*

example

- Consider a very simple agent: the square-root function
- on a calculator. The percept sequence for this agent is a sequence of keystrokes representing anumber, and the action is to display a number on the display screen. The ideal mapping is that
- when the percept is a positive number *x, the right action is to display a positive number z such*
- that z2 « *x, accurate to, say, 15 decimal places.*

Example ideal mapping for the square-root problem (accurate to 1 5 digits), and a corresponding program that implements the ideal mapping.

Percept *x*	Action	
1.0 1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 . . .	1.000000000000000 1.048808848170152 1.095445115010332 1.140175425099138 1.183215956619923 1.224744871391589 1.264911064067352 1.303840481040530 1.341640786499874 1 .378404875209022	function SQRT(X) z <— 1 .0 / * *initial guess * 1* repeat until]z2 - *x] < 10~15* z <- z - *(z2 - x)/(2z)* end return z Figure

Autonomy

- *A system is*
- *autonomous to the extent that its behaviour is determined b\ its own experience.*

STRUCTURE OF INTELLIGENT AGENTS

- The job of AI is to design the **agent program**
- It is a function that implements the agent mapping from precepts to actions.
- We assume this program will run on some sort of computing device, which we will call the **architecture.**
- The relationship among agents, architectures, and programs can be summed up as follows:
- *agent = architecture + program*

Structure of Intelligent Agents

Agent Type	Precepts	Actions	Goals	Environment
Medical diagnosis system	Symptoms, findings, patient's answers	Questions, tests, treatments	Healthy patient, minimize costs	Patient, hospital
Satellite image analysis system	Pixels of varying intensity, color	Print a categorization of scene	Correct categorization	Images from orbiting satellite
Part-picking robot	Pixels of varying intensity	Pick up parts and sort into bins	Place parts in correct bins	Conveyor belt with parts
Refinery controller	Temperature, pressure readings	Open, close valves; adjust temperature	Maximize purity, yield, safety	Refinery
Interactive English tutor	Typed words	Print exercises, suggestions, corrections	Maximize student's score on test	Set of students

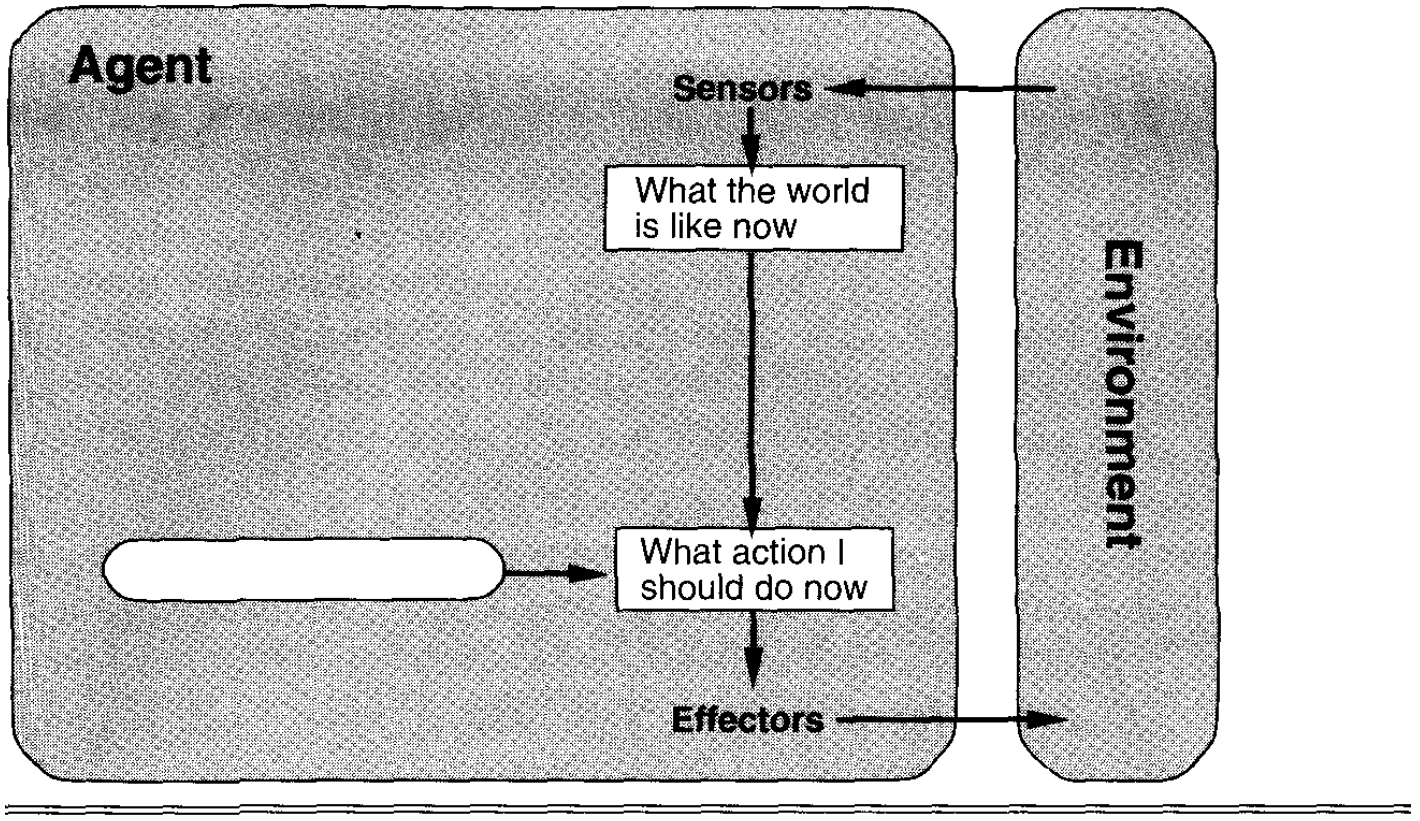
Agent programs

- **function *SKELETON-AGEN~[(percept) returns action***
- **static: *memory, the agent's memory of the world***
- *memory — UPDATE-MEMORY(memory, percept)*
- *action <— CHOOSE-BEST-ACTION(memory)*
- *memory — UPDATE-MEMORY(/memory action)*
- **return *action***
- A skeleton agent. On each invocation, the agent's memory is updated to reflect
- the new percept, the best action is chosen, and the fact that the action was taken is also stored in memory.
- The memory persists from one invocation to the next

An example

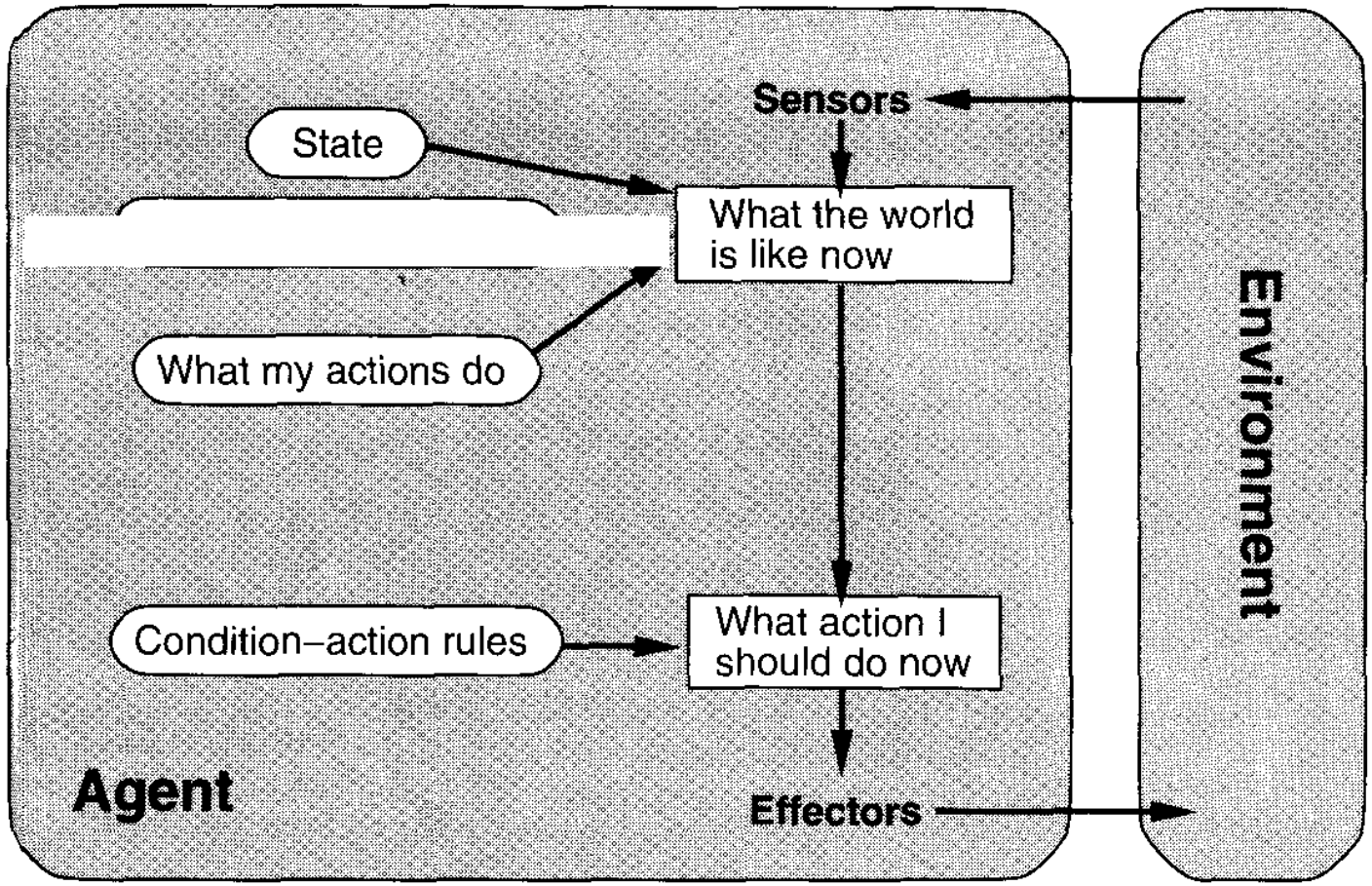
Agent Type	Precepts	Actions	Goals	Environment
Taxi driver	Cameras, speedometer, GPS, sonar, microphone	Steer, accelerate, brake, talk to passenger	Safe, fast, legal, comfortable trip, maximize profits	Roads, other traffic, pedestrians, customers

Structure of Intelligent Agents 41 Condition-



- different types of agent program. We will consider four types of agent program:
- • Simple reflex agents
- • Agents that keep track of the world
- • Goal-based agents
- • Utility-based agents

SIMPLE REFLEX AGENT



- A simple reflex agent works by finding a rule whose condition matches thecurrent situation (as defined by the percept) and then doing the action associated with that rule.

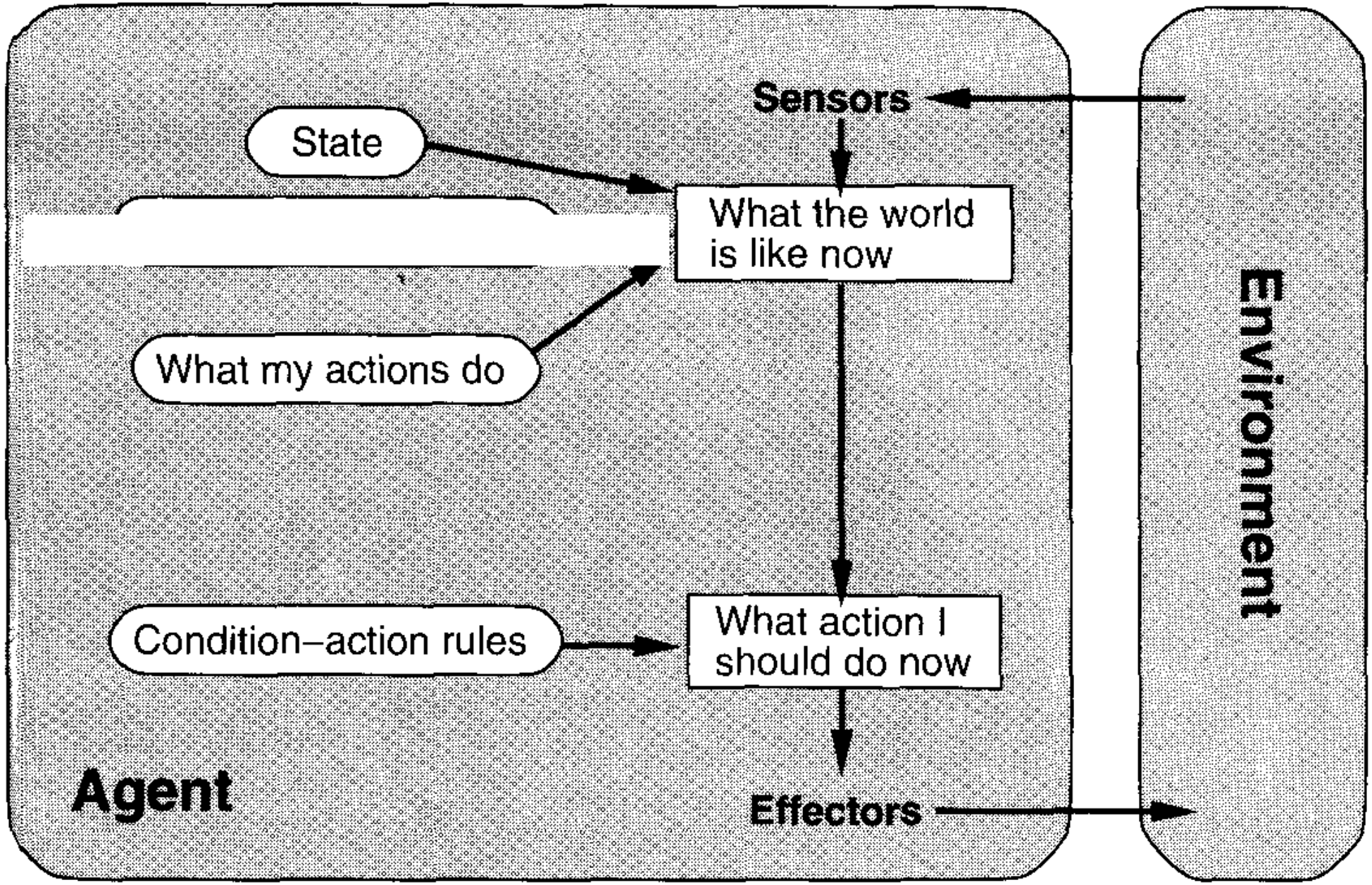
- **function SiMPLE-REFLEX-AGENT(percept) returns *action***
- **static: *rules, a set of condition-action rules***
- *state <— lNTERpRET-lNPUT(***percept)**
- rule<- RULE-MATCH(.state, *rules)*
- *action <- RULE-AcTiON(rule)*
- **return *action***
- Example
- **if *car-in-front-is-braking then initiate-braking***

Agents that keep track of the world

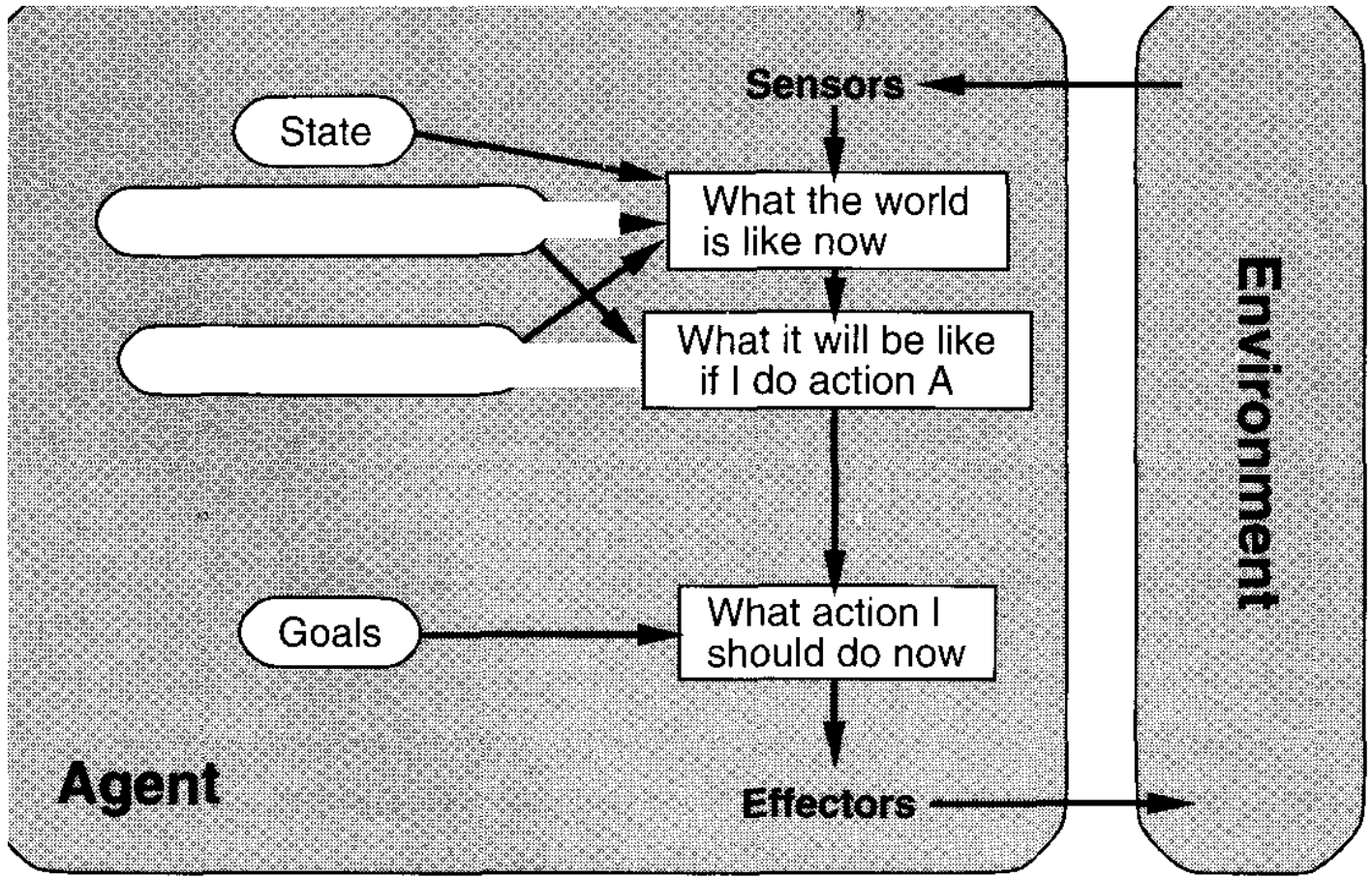
- **function REFLEX-AGENT-WiTH-STATE(percept) returns action**
- **static: *state, a description of the current world state***

rules, a set of condition-action rules

- *state <— UPDATE-STATE(.state, percept)*
- *rule — RULE-MATCH(state, rules)*
- *action — RULE-ACTION[rule]*
- *state <- UPDATE-STATE(state, action)*
- **return *action***
- A reflex agent with internal state. It works by finding a rule whose condition
- matches the current situation (as defined by the percept and the stored internal state) and thendoing the action associated with that rule.


Sensors
State
What the world is like now
What my actions do
Condition–action rules
What action I should do now
Agent
Effectors
Environment

Goal-based agents



- the agent needs some sort of **goal information**
- The agent program can combine this with information about the results of possible actions (the same information as was used to update internal state in the reflex agent) in order to choose actions that achieve the goal. Sometimes j
- this will be simple, when goal satisfaction results immediately from a single action; sometimes, it will be more tricky, when the agent has to consider long sequences of twists and turns to find away to achieve the goal. **Search and planning**
- Search and planning are the subfields of AI devoted to finding action sequences that do achieve the agent's goals.

Utility-based agents

- Goals alone are not really enough to generate high-quality behavior
- Utility is a function that maps a state onto a real number,
- A complete specification of the utility function allows rational
- decisions in two kinds of cases where goals have trouble. First, when there are conflicting goals, only some of which can be achieved (for example, speed and safety),
- the utility function specifies the appropriate trade-off..

ENVIRONMENTS

- we will see how to couple an agent to an environment.

- we will describe the different types of environments and how they affect the design of agents.

Properties of environments

- Environments come in several flavors. The principal distinctions to be made are as follows:
- **Accessible vs. inaccessible.**
- If an agent's sensory apparatus gives it access to the complete state of the environment,
- Then we say that the environment is accessible to that agent.
- **Deterministic vs. Nondeterministic**
- If the next state of the environment is completely determined by the current state and the actions selected by the agents, then we say the environment is deterministic. In principle,
- an agent need not worry about uncertainty in an accessible,

- **Episodic vs. Nonepisodic**
- In an episodic environment, the agent's experience is divided into "episodes." Each episode consists of the agent perceiving and then acting.
- The quality of its action depends just on the episode itself, because subsequent episodes do not depend on what actions occur in previous episodes.
- Episodic environments are much simpler because the agent does not need to think ahead.

Static vs. Dynamic

- If the environment can change while an agent is deliberating, then we say the environment is dynamic for that agent; otherwise it is static.
- Static environments are easy to deal with because the agent need not keep looking at the world while it is deciding on an action,

- **Discrete vs. continuous.**
- If there are a limited number of distinct, clearly defined percepts and actions we say thatthe environment is discrete. Chess is discrete—there are a fixed number of possible moveson each turn.
- Taxi driving is continuous—the speed and location of the taxi and the other vehicles sweep through a range of continuous values

Environment	Accessible	Deterministic	Episodic	Static	Discrete
Chess with a clock	Yes	Yes	No	Semi	Yes
Chess without a clock	Yes	Yes	No	Yes	Yes
Poker	No	No	No	Yes	Yes
Backgammon	Yes	No	No	Yes	Yes
Taxi driving	No	No	No	No	No
Medical diagnosis system	No	No	No	No	No
Image-analysis system	Yes	Yes	Yes	Semi	No
Part-picking robot	No	No	Yes	No	No
Refinery controller	No	No	No	No	No
Interactive English tutor	No	No	No	No	Yes

Environment programs

- **procedure RuN-ENViRONMENT(>tefc, UPDATE-FN, *agents, termination)***
- **inputs: *state, the initial state of the environment***
- UPDATE-FN, function to modify the environment
- *agents, a set of agents*
- *termination, a predicate to test when we are done*
- **repeat**
- **for each *agent in agents do***
- *PERCEPT[agent] <— GEY-PERCEPT(agent, state)*
- **end**
- **for each *agent in agents do***
- *ACTlON[agent] — PROGRAM[agent](PERCEPT[agent})*
- **end**
- *state <— UPDATE-FN(acft'o«.s, agents, state)*
- **until *termination(state)***

Chapter 3
SOLVING PROBLEMS BY SEARCHING

- goal-based agent called a **problem-solving agent.**
- Problem-solving agents decide what to do by finding sequences of actions that lead to desirable states.
- **Goal formulation, based on the current situation,** is the first step in problem solving.
- **Problem formulation is the process of deciding what actions and states to consider, and follows goal** formulation.
- This process of looking for such a sequence is called **search. A search algorithm takes** a problem as input and returns a **solution in the form of an action sequence**
- The actions it recommends can be carried out. This is called the **execution phase**

A simple problem-solving agent

- **function SIMPLE-PROBLEM-SOLVING-AGENT(P) returns an action**
- **inputs: *p, a percept***
- **static: *s, an action sequence, initially empty***
- *nitialstate, some description of the current world state*
- *g, a goal, ily null*
- *problem, a problem formulation*
- *state <— UPDATE-STATE(state, p)*
- if *s is empty then*
- g — FORMULATE-GOAL(,stafe)
- *problem <— FORMULATE-PROBLEM(.state, g)*
- *s <- SEARCH(problem)*
- *action — RECOMMENDATION(s, state)*
- *s <— REMAINDER(s, state)*
- **return *action***

FORMULATING PROBLEMS

- there are four essentially different types of problems—
- Single state problems,
- multiple-state problems,
- contingency problems, and
- exploration problems

- **Knowledge and problem types**
- **single-state problem**
- **Only one state**
- Forexample, if its initial state is 5, then it can calculate that the action sequence *[Right,Suck] will get* to a goal state. This is the simplest case, which we call a **single-state problem.**
- **multiple-state problem.**
- **contingency problem.**
- **Interleaving problem**
- **exploration problem**

Well-defined problems and solutions

- **A problem is really a collection of information that the agent will use to decide what to do. We** will begin by specifying the information needed to define a single-state problem.
- We have seen that the basic elements of a problem definition are the states and actions

INITIAL STATE

- **The initial state that the agent knows itself to be in.**

OPERATOR

- The set of possible actions available to the agent. The term **operator is used to denote** the description of an action in terms of which state will be reached by carrying out the action in a particular state.

SUCCESSOR FUNCTION

- (An alternate formulation uses a **successor function *S. Given***

a particular state *x, S(x) returns the set of states reachable from x by any single action.)*

STATE SPACE

- These define the **state space of the problem: the set of all states reachable from the** initial state by any sequence of actions

- PATH
- A **path in the state space is simply any sequence of** actions leading from one state to another.
- GOAL TEST
- **The goal test, which the agent can apply to a single state description to determine if it is** a goal state. Sometimes there is an explicit set of possible goal states, and the test simply checks to see if we have reached one of them.
- PATH COST
- **A path cost function is a function that assigns a cost to a path. In all cases we will consider,** the cost of a path is the sum of the costs of the individual actions along the path. The path cost function is often denoted by *g.*

SOLUTION

- The output of a search algorithm is a **solution,**

SEARCHING FOR SOLUTIONS

- The idea is to maintain and extend a set of partial solution sequences.
- we show how to generate these sequences and how to keep track of them using suitable data structures.
- **Generating action sequences**
- This is done by applying the operators ,GENERATING to the current state, thereby **generating a new set of states. The process is called expanding the** state.
- search process as building up a **search tree that is superimposed**
- SEARCH NODE over the state space. The root of the search tree is a **search node.**

Data structures for search trees

- we will assume a node is a data
- structure with five components:
- • the state in the state space to which the node corresponds;
- • the node in the search tree that generated this node (this is called the **parent node);**
- • the operator that was applied to generate the node;
- • the number of nodes on the path from the root to this node (the **depth of the node);**
- • the path cost of the path from the initial state to the node

- we will assume that the collection of nodes is a implemented as a **queue. The**
- operations on a queue are as follows:
- • *MAKE-Q\JEUE(Elements) creates a queue with the given elements.*
- • EMPTY?*'(Queue) returns true only if there are no more elements in the queue.*
- • REMOVE-FRONT(gweMe) removes the element at the front of the queue and returns it.
- • *QUEUlNG-FN(Elements,Queue) inserts a set of elements into the queue. Different varieties*
- of the queuing function produce different varieties of the search algorithm.

SEARCH STRATEGIES

- > **Completeness: is the strategy guaranteed to find a solution when there is one?**
- **Time complexity: how long does it take to find a solution?**
- **Space complexity: how much memory does it need to perform the search?**
- **Optimality: does the strategy find the highest-quality solution when there are several**
- different solutions?7

Uninformed search/blind search

- The term means that they have no information about the number of steps or the path cost from the current state to the goal—all they can do is distinguish a goal state from a non goal state.
- Uninformed search is also sometimes called **blind search.**

informed search strategies or heuristic search

- An uninformed search has no preference among these, but a more clever agent might notice that the goal,
- Bucharest, is southeast of Arad, and that only Sibiu is in that direction, so it is likely to be the best choice

Breadth-first search

- One simple search strategy is a **breadth-first search**
- In this strategy, the root node is expanded
- first, then all the nodes generated by the root node are expanded next, and then *their successors,*
- and so on.
- **branching factor of these states (and of the search tree) is *b.***
- The root of the search tree generates *b nodes at the first level, each of which generates b more* nodes, for a total of *b2 at the second level. Each of these generates b more nodes, yielding b3* nodes at the third level, and so on. Now suppose that the solution for this problem has a path
- length of *d. Then the maximum number of nodes expanded before finding a solution is*
- $1 + b + b2 + b3 + \bullet \bullet \bullet + bd$

INFORMED SEARCH METHODS

- uses problem-specific knowledge
- BEST-FIRST SEARCH
- the knowledge to make this determination is provided by **evaluation function**
- Best first search, uses an evaluation function to decide which among the various available nodes is the most promising (or ‘BEST’) before traversing to that node.

UNIT II:KNOWLEDGE AND REASONING

- A logical, knowledge-based agent begins with some knowledge of the world and of its own actions.
- It uses logical reasoning to maintain a description of the world as new precepts arrive, and to deduce a course of action that will achieve its goals.
- these basic elements work in a logic called **propositional logic,**

A KNOWLEDGE-BASED AGENT KNOWLEDGE

- KNOWLEDGE BASE
- The central component of a knowledge-based agent is knowledge base, or KB. Informally, a knowledge base is a set of representations of facts about the world.
- SENTENCE
- Each individual representation SENTENCE " is called a sentence.
- knowledge representation language.
- The sentences are expressed in a language called a knowledge representation language.

- The standard names for these tasks are TELL and ASK,
- Determining what follows from what the KB has been TELLed is the job of the inference mechanism, the other main component of a knowledge-based agent
- **Background knowledge**
- Each time the agent program is called, i tdoes two things. First, it TELLS the knowledge base what it perceives.1 Second, it ASKS the
- knowledge base what action it should perform. In the process of answering this query, logical reasoning is used to prove which action is better than all others, given what the agent knows and
- what its goals are. The agent then performs the chosen action.

- **function *KB-AGEN1(percept) returns an action***
- static: *KB, a knowledge base*
- *t, a counter, initially 0, indicating time*
- TELL(A'S, MAKE-PERCEPT-SENTENCE(percepf, *t))*
- *action — ASK(KB, MAKE-ACTION-QUERY(f))*
- TELL(A"B, MAKE-ACTION-SENTENCE(flCf(0«, *t))*
- *T-t+ 1*
- **return *action***

Challenges for AI

- Better understanding of our brain capacity
- Better use of computer capacity
- **Data privacy and security:** Most AI applications rely on huge
- volumes of data to learn and make intelligent decisions

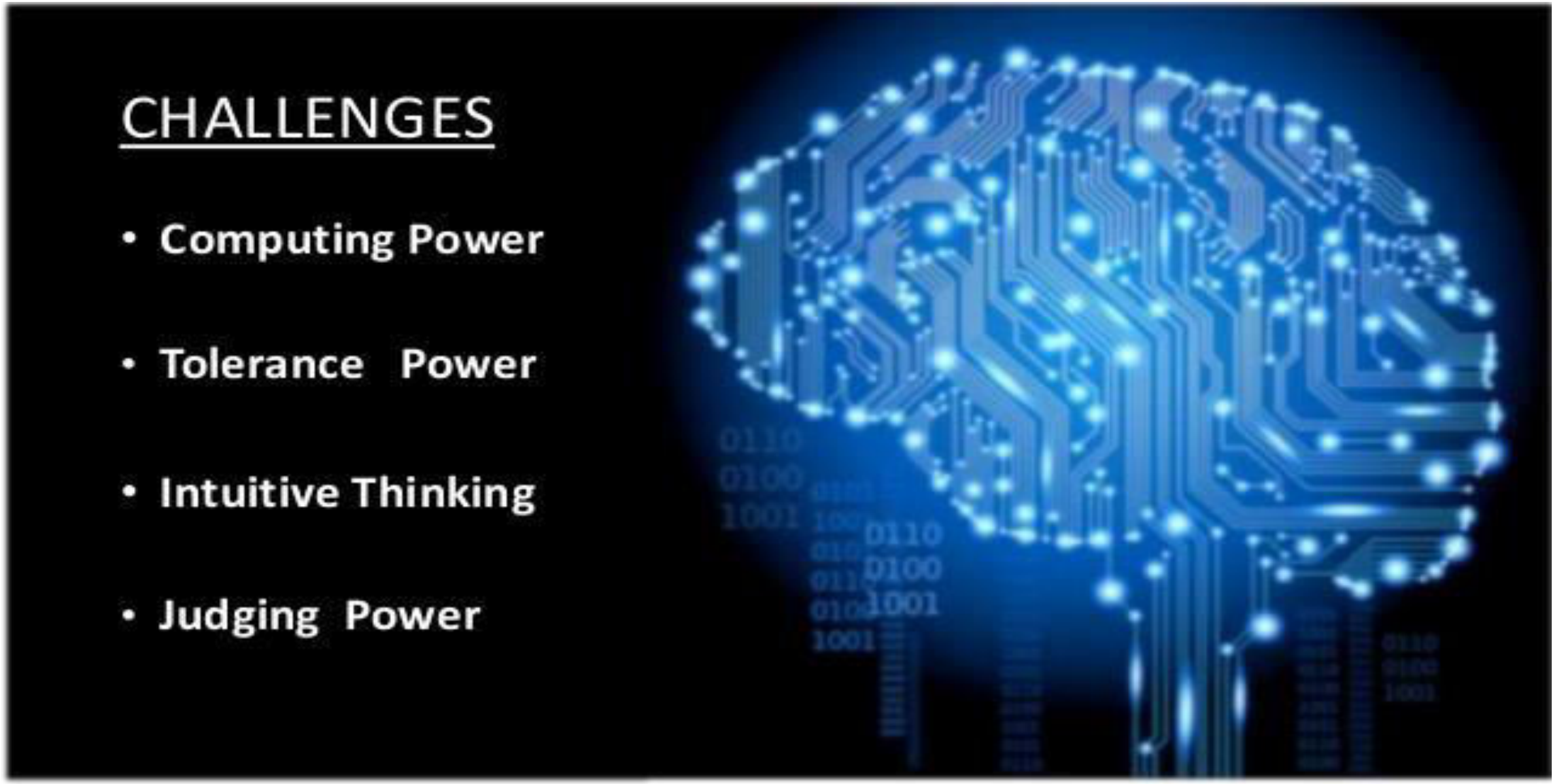
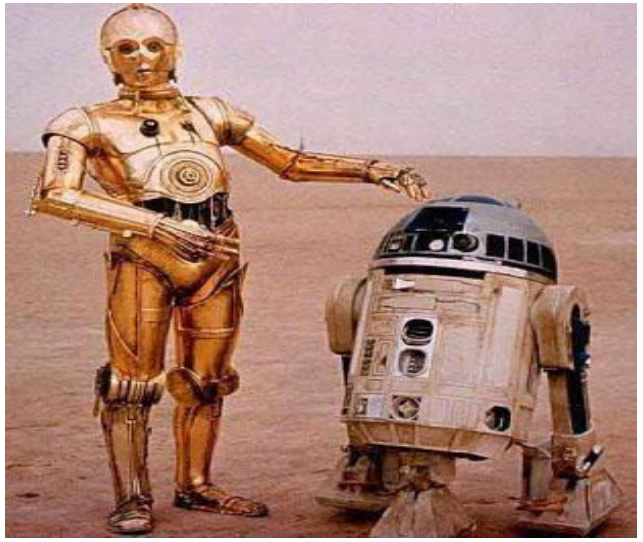


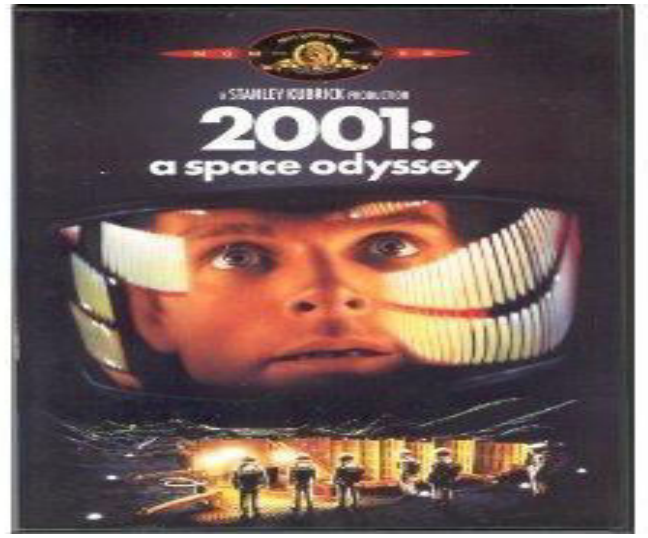
Fig.2. Challenging Issue in AI

Applications Areas

Fig.3. Applications of AI

Artificial Intelligence in Movies





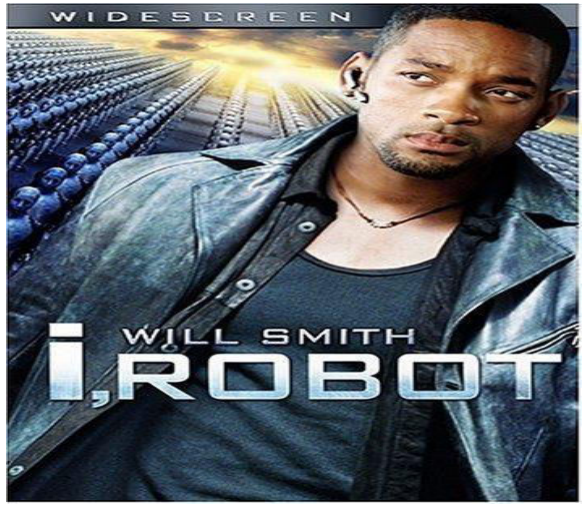


Fig.4. AI in Graphics

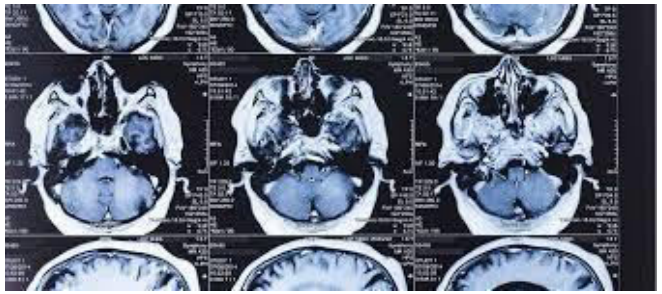
Artificial Intelligence in Real Life

A young *science* (≈ 50 years old)

- Impressive success stories
- "Intelligent" in specialized domains
- Many application areas

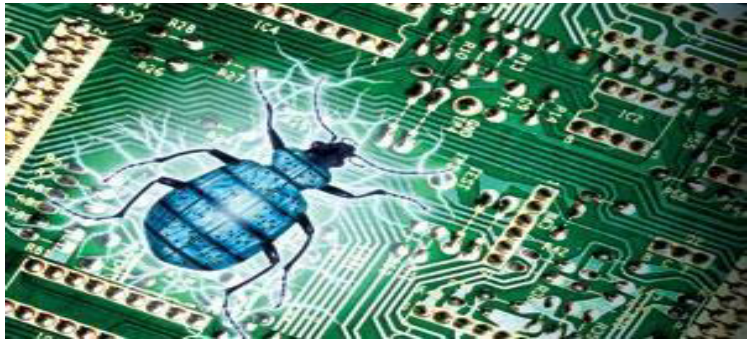
Industry

Gaming

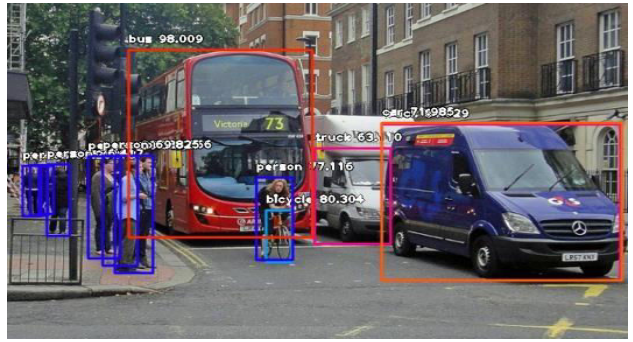


Medical Domain

Face detection



Formal verification



Detection and Tracking

Fig.5 Real time Applications

Smart cars

- Self-driving cars are becoming increasingly a reality with each passing moment.
- For Eg. Google's self-driving car project (on going)

Fig 6.Self Driving Car: Google Project

Object Detection and Tracking

- Moving object or human detection in Indoor or Outdoor Scenario’s

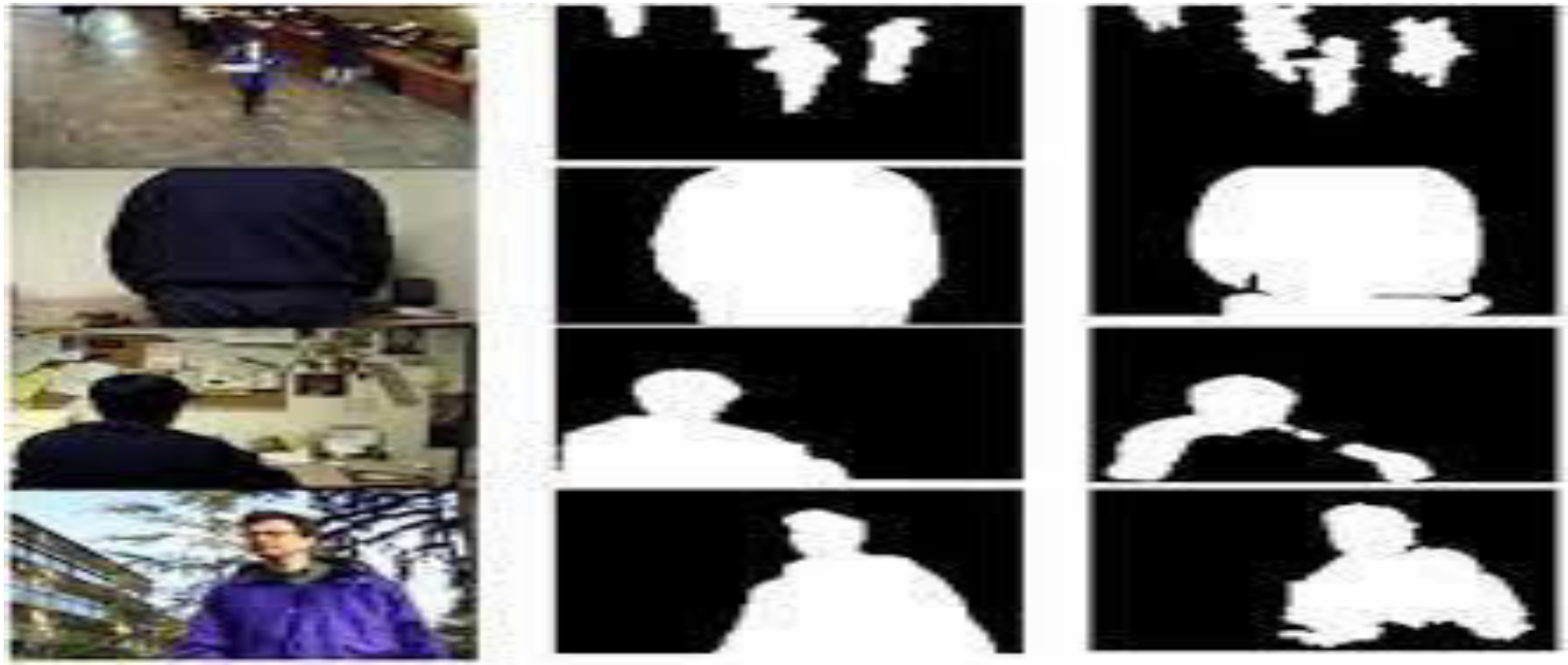


Fig.7. Moving object detection in Indoor or outdoor Video sequence

Robots

- Excel at performing simple, repetitive tasks
- Free workers from tedious or hazardous jobs
- Have limited mobility
- Operation is controlled by a computer program that includes commands
- Includes programming languages for controlling
    - Variable Assembly Language (VAL)

Honda Humanoid Robot

Walk

Turn

Stairs

Fig.8. Robotics

Fig.9. Military Robots

Expert Systems

- Mimics human expertise in a field to solve a problem in a well-defined area
- Used for activities that human experts have already handled successfully
  - Tasks in medicine, geology, education, and oil exploration
    - For E.g. Dendril and Mycin

Surveillance

- AI can be trained using supervised exercises, security algorithms to take input from security cameras.
- Eventually, they can identify potential threats and warn human security officers to investigate further.

Preserving Wildlife

- Wildlife preservation is notoriously difficult, especially when attempting to analyze population sizes or track animals.
- Scientists, simply, cannot possibly track every animal or tag them all with GPS devices.
- A team in Chicago have successfully implemented a form of AI, developed by Wildbrook.org to track animals

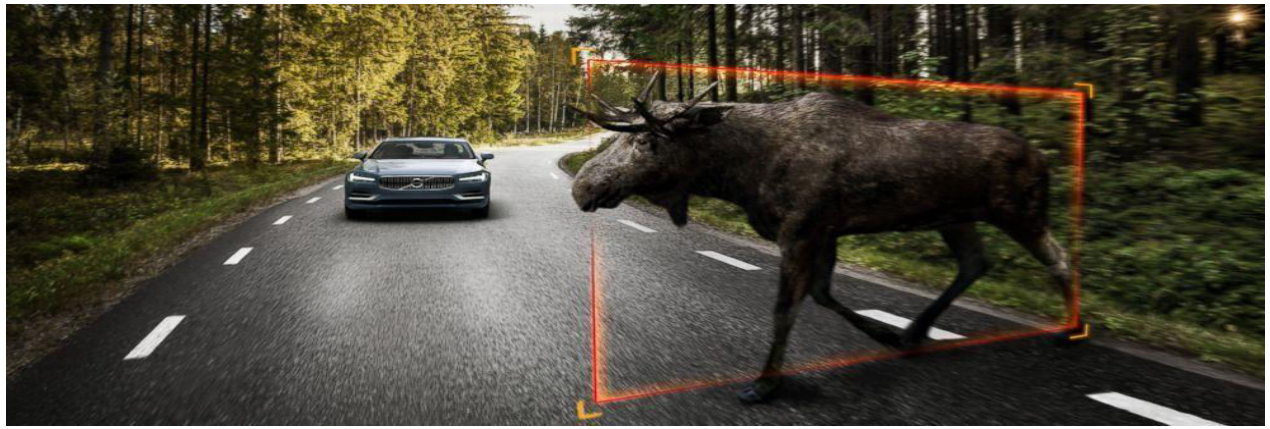


Fig.9. Wildlife Surveillance

Finance

- Banks use artificial intelligence systems to organize operations, invest in stocks, and manage properties
- Loan investigation, ATM design, safe and fast banking etc. also uses AI.
- In August 2001, robots beat humans in a simulated financial trading competition

Hospitals and medicine

- A medical clinic can use artificial intelligence systems to organize bed schedules, make a staff rotation, and provide medical information and other important tasks.
- AI has also applications in field of cardiology (CRG), Neurology (MRI), Embryology (sonography), complex operations of internal organs etc.

Heavy industry

- Huge machines involves risks in manual maintenance and working .
- Robots are safe and efficient agent for it.
- They are often given jobs that are considered dangerous to humans.
- Robots have proven effective in jobs that are very repetitive which may lead to mistakes or accidents due to a lapse in concentration and other jobs which humans may find degrading

Gaming

- AI has also been applied to video games, for example video game bots, which are designed to stand in as opponents where humans aren't available or desired

Consumer Marketing

- Have you ever used any kind of credit/ATM/store card while shopping?
  - if so, you have very likely been “input” to an AI algorithm
- All of this information is recorded digitally
- Companies like Nielsen gather this information weekly and search for patterns
  - general changes in consumer behavior
  - tracking responses to new products

Identification Technologies

- ID cards
  - e.g., ATM cards
  - can be a nuisance and security risk:
    - cards can be lost, stolen, passwords forgotten, etc
      - Biometric Identification
        - walk up to a locked door
    - camera
    - fingerprint device
    - microphone
  - computer uses your biometric signature for identification
    - face, eyes, fingerprints, voice pattern

References

- Lavanya Sharma, “Introduction: From Visual Surveillance to Internet of Things”, From Visual Surveillance to Internet of Things”, Taylor & Francis, CRC Press, Vol.1, pp.14
- Shubham, Shubhankar, Mohit, Lavanya Sharma, “Use of Motion Capture in 3D Animation: Motion Capture Systems, Challenges, and Recent Trends”, in 1st IEEE International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (Com-IT-Con), India, pp. 309-313, 14th -16th Feb
- Sharma, L. (Ed.), Garg, P. (Ed.). (2020). From Visual Surveillance to Internet of Things.New York: Chapman and Hall/CRC, https://doi.org/10.1201/9780429297922
- Rich and Knight, “Artificial Intelligence”, Tata McGraw Hill, 1992
- S. Russel and P. Norvig, “Artificial Intelligence – A Modern Approach”, Second Edition, Pearson Edu.
- KM Fu, "Neural Networks in Computer Intelligence", McGraw Hill
- Russel and Norvig, "Artificial Intelligence: A modern approach", Pearson Education