INTELLIGENT AGENT

- An agent is anything that can be viewed as **perceiving** its environment through **sensors** and **acting** upon that environment through **effectors**.

- **Examples** (human agent, software agent, robotic agent):

- A **human agent** has eyes, ears, and other organs for sensors, and hands, legs, mouth, and other body parts for effectors.

- A **robotic agent** substitues cameras and infrared range finders for the sensors and various motors for the effectors.

- A **software agent** has encoded bit strings as its percepts and actions (e.g. game playing agent)

MAPOWANIE (BODZIEC → AKCJA)

Idealne mapowanie – wyspecyfikowanie jakie działania (względem środowiska) powinny być podjęte w odpowiedzi na zaobserwowany bodziec (sekwencję bodźców).

Bodziec x	Akcja z
1.0	1.00000
1.1	1.04880
1.2	1.09544
1.3	1.14017
1.4	1.18321
1.5	1.22474
1.6	1.26491
1.7	1.30384
1.8	1.37840
...	...

function SQRT(x)
z := 1.0
repeat until $|z^2 - x| < 10\ -15$
    $z := z - (z^2 - x) / (2\ z)$
end
return z

RATIONAL AGENT

- **Rational agent** - four components:
- ***percept sequence*** (everything that the agent has perceived so far)
- ***actions*** (what the agent can perform)
- ***built-in-knowledge*** (what the agent knows about the environment / world)
- ***performance measure*** (defines the degree of success, must be an objective measure - by outside observer - independent of agent's own thoughts about its behaviour) – **example**
- **RATIONALITY vs. OMNISCIENCE**

AUTONOMOUS AGENT

- If the agent’s actions are based completely on built-in-knowledge (i.e. does not need to pay attention to its percepts), then we say that the agent lacks **autonomy**.

- **A system is autonomous to the extent that its behavior is determined by its own experience ➔**
  *** it is able to improve its preformance by learning ***

- **A truly autonomous intelligent agent should be able to operate successfully in a wide variety of environments, given sufficient time to adapt.**

PAGE DESCRIPTION

Agent type	Percepts	Actions	Goals	Environment
Medical diagnosis system	Symptoms, findings, patient's answers	Questions, tests, treatments	Healthy patient, minimize cost	Patient, hospital
Satellite image analysis system	Pixels of varying intensity, color	Print a categorization of a scene	Correct categorization	Images from orbiting satelite
Part-picking robot	Pixels of varying intensity, color	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	Group of students

AGENT PROGRAM

- **function** SKELETON_AGENT(*percept*)
- **static**: *memory*
-
- *memory* := UPDATE_MEMORY(*memory, percept*)
- *action* := CHOOSE_BEST_ACTION(*memory*)
- *memory* := UPDATE_MEMORY(*memory, action*)

- **return** *action*

- CHOOSE_BEST_ACTION is defined as a function from **percept sequences** to **actions.** The agent program receives only a single percept as its input. It is up to the agent to build up the percept sequence in memory, if desired.
- PERFORMANCE MEASURE ...

LOOKUP TABLE. WHY NOT ?

- **function** TABLE_DRIVEN_AGENT(*percept*)
- **static**: *percepts*
- *table*
-
- append *percept* to the end of *percepts*
- *action* := LOOKUP(*percepts, table*)
- **return** *action*
- **PROBLEMS?**
- A table-driven agent **does** what we want, but it is not the desirable way to do that (it is not „intelligent”).

FOCUS EXAMPLE: AUTONOMOUS TAXI DRIVER

Agent type	Percepts	Actions	Goals	Environment
Taxi driver	Cameras, odometer, speedometer, GPS, sonar, microphone or keyboard	Steer, accelerate, brake, talk to passenger	Safe, fast, legal, comfortable trip, maximize profits	Road, other traffic, pedestrians, customers

performance measure: getting to correct destination, minimizing fuel consumption and wear, minimizing the time and/or cost, minimizing violations of traffic laws, maximizing safety of the passenger, their comfort.

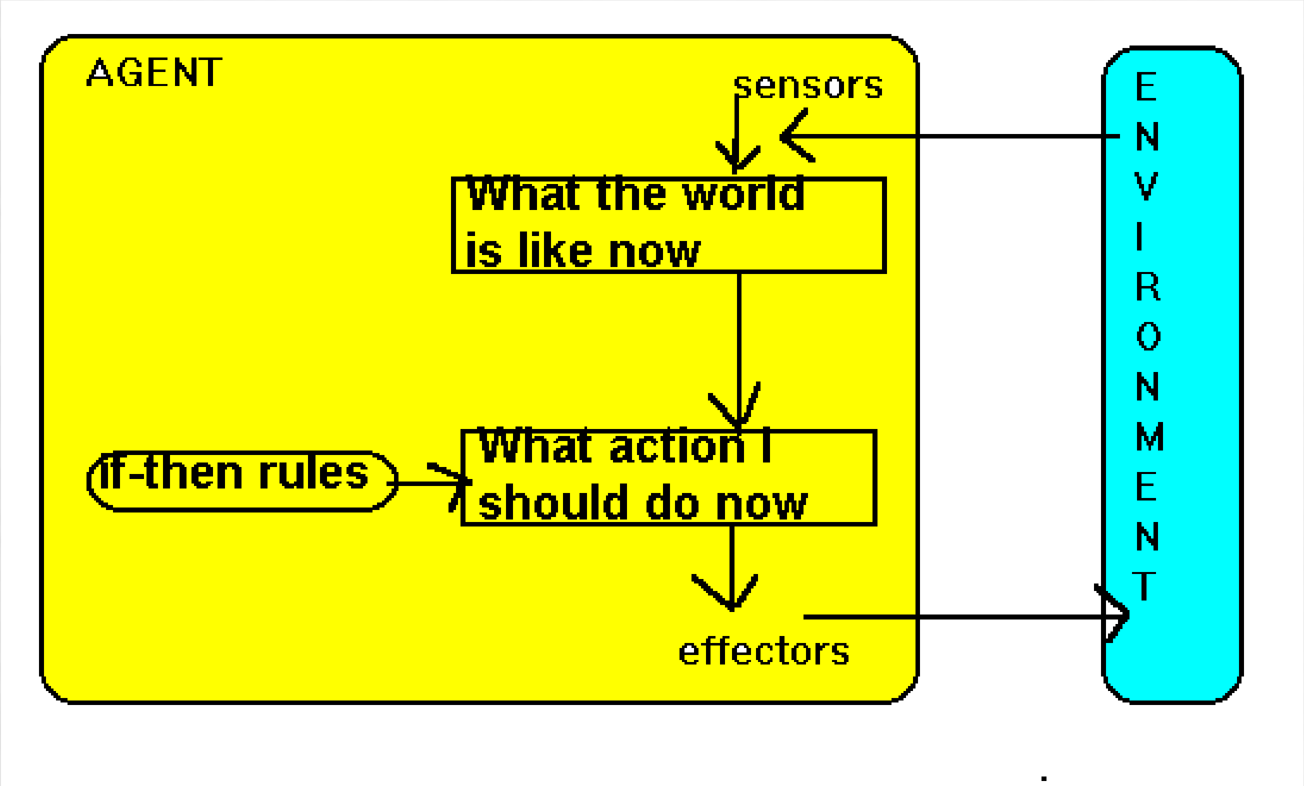
Some of the goals are conflicting, therefore there must be a trade-off involved.

LET'S LOOK AT 4 TYPES OF AGENTS

- **Four types of agent programs:**

- Simple reflex agent
- Agent that keeps track of the world
- Goal-based agent
- Utility-based agent

SIMPLE REFLEX AGENT



- Such agents can be implemented very efficiently, but the range of their applicability is narrow.

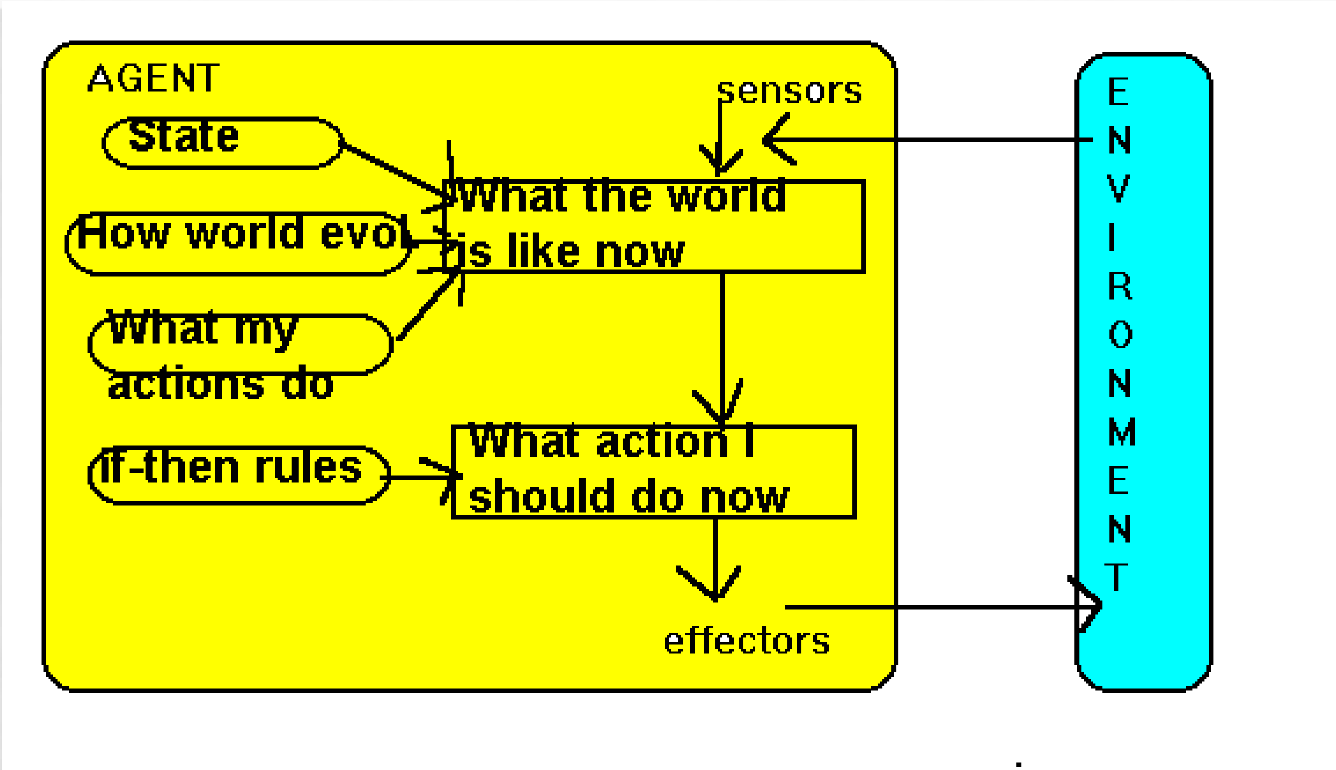
AGENT THAT KEEPS TRACK OF THE WORLD

- There is need for the internal (memory) state in order to distinguish between the two the same states but with different percept sequence and which require different reactions.

- Example1: check whether the brake lights are on - compare with the previous picture

- Example2: a car which is going on the next line in invisible --> the situation the same as it were not there. Different reaction required can/cannot change line !!!

AGENT THAT KEEPS TRACK OF THE WORLD



- Function UPDATE_STATE is responsible fro creating a new internal state description, as well as interpreting the new percept. It uses information about how the world evolves to keep track of the unseen parts of the world, and also must know about what the agent actions do to the state of the world.

GOAL-BASED AGENT

- Knowing about the current state of the environment is not always enough to decide what to do.
- **Example:** a taxi at a cross-roads. Needs to know the destination (the goal) in order to turn right/left or go straight on. **The goal is: being at the passenger's destination.**

- **Goal-based agent** asks questions: what will happen if I do such and such ? Will that make me happy ? The reflex agent brakes when it sees the brake lights. A goal-based agent, in principle can reason in the following way: if the car in front has its brake lights on, it will slow down.
- The only action that the agent can take to reach the destination safely is to slow down, too !!!

- It appears to be less efficient than the reflex agent but it is much more flexible. For example, by specifying a new destination we can get the goal based agent to come up with a new behavior. The relex agent will work only for one destination. For different one it requires rewritting the rules about where to turn, and so on.

UTILITY-BASED AGENT

- Goals alone are not really enough to generate high-quality behavior. For example, there are many action sequences that will get the taxi to its destination, thereby achieving a goal but some are quicker, safer more reliable or cheaper than others.
- Goals just provide a crude distinction between „happy" „unhappy" states, whereas a more general performance measure should allow a comparison between different world states (how happy am I?). If one world is preferred to another then it has higher **utility** for an agent.

- **Utility is a function that maps a state onto a real number, which describes the associated degree of happiness.**

- **Comments:**
- **-** if there are conflicting goals utility help with the appropriate trade-off
- \- when there are several goals that the agent can aim for, the utility specifies the way in which the likelihood of the goal can be weighted against the importance of the goal.

ENVIRONMENTS (PAGE)

- **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 the 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.
- **episodic** vs. **nonepisodic**: 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 action occur in previous episodes. Episodic environments are much simpler because the agent does not need to think ahead.
- **static** vs. **dynamics (semidynamic)**: if the environment can change while the agent is deliberating then we say the environment is dynamic for that agent; otherwise it is static. If the environment does not change with the passage of time, but the agent's performance score does, then we say the environment is **semidynamic**.
- **discrete** vs. **continuous:** if there are a limited number of distinct, clearly defined percepts and actions we say taht the environment is discrete.

- **The hardest environments: inaccessible, nondetreministic, nonepisodic, dynamic and continuous.**

ENVIRONMENTS - EXAMPLES

Environment	Accessible	Detrministic	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

INTELLIGENT AGENTS - SUMMARY

- An **agent** is something that *perceives* and *acts* in an **environment**. It can be split into an **architecture** and an **agent program**.

- An agent is **autonomous** to the extent that its action choices depend on its own experience, rather than on knowledge of the environment that has been built-in by the designer.

- An **agent program** maps from a *percept* to an *action*, while updating an **internal state**.

INTELLIGENT AGENTS - SUMMARY

- The appropriate design of the **agent program** depends on the ***p**ercepts*, ***a**ctions, **g**oals*, and ***e**nvironment (**page**)*.

- **Reflex agents** respond immediately to percepts, **goal-based agents** act so that they will achieve their goal(s), and **utility-based agents** try to maximize their own "happines".

- The process of making decisions by *reasoning with knowledge* is central to AI and to successful agent design, therefore, *representing knowledge* is important.

- Some **environments** are more demanding than others. **Environments** that are **inaccessible**, **nondeterministic**, **nonepisodic**, **dynamic**, and **continuous** are the most challenging.