How do you design an intelligent agent?

- **Intelligent agents** perceive environment via **sensors** and act rationally on them with their **effectors**
- Discrete agents receive **percepts** one at a time, and map them to a sequence of discrete **actions**
- **Properties**
  - Reactive to the environment
  - Pro-active or goal-directed
  - Interacts with other agents through communication or via the environment
  - Autonomous

**A specific example: Automated taxi driving system**

- **Percepts**: Video, sonar, speedometer, odometer, engine sensors, keyboard input, microphone, GPS, ...
- **Actions**: Steer, accelerate, brake, horn, speak, ...
- **Goals**: Maintain safety, reach destination, maximize profits (fuel, tire wear), obey laws, provide passenger comfort, ...
- **Environment**: U.S. urban streets, freeways, traffic, pedestrians, weather, customers, ...
- Different aspects of driving may require different types of agent programs!

**sensors/percepts and effectors/actions?**

- **Humans**
  - Sensors: Eyes (vision), ears (hearing), skin (touch), tongue (gustation), nose (olfaction), neuromuscular system (proprioception)
  - Percepts:
    - At the lowest level – electrical signals from these sensors
    - After preprocessing – objects in the visual field (location, textures, colors, ...), auditory streams (pitch, loudness, direction), ...
  - Effectors: limbs, digits, eyes, tongue, ...
  - Actions: lift a finger, turn left, walk, run, carry an object, ...
- **The Point**: percepts and actions need to be carefully defined, possibly at different levels of abstraction
Rationality

• An ideal rational agent should, for each possible percept sequence, do actions to maximize its expected performance measure based on
  (1) the percept sequence, and
  (2) its built-in and acquired knowledge.
• Rationality includes information gathering, not “rational ignorance” (If you don’t know something, find out!)
• Rationality → Need a performance measure to say how well a task has been achieved.
• Types of performance measures: false alarm (false positive) and false dismissal (false negative) rates, speed, resources required, effect on environment, etc.

Autonomy

• A system is autonomous to the extent that its own behavior is determined by its own experience.
• Therefore, a system is not autonomous if it is guided by its designer according to a priori decisions.
• An autonomous agent can always say “no”.
• To survive, agents must have:
  – Enough built-in knowledge to survive.
  – The ability to learn.

Some agent types

(0) Table-driven agents
  – use a percept sequence/action table in memory to find the next action. Implemented by a (large) lookup table
(1) Simple reflex agents
  – Based on condition-action rules, implemented with an appropriate production system; stateless devices with no memory of past world states
(2) Agents with memory
  – have internal state that is used to keep track of past states of the world
(3) Agents with goals
  – Agents that have state and goal information that describes desirable situations. Agents of this kind take future events into consideration.
(4) Utility-based agents
  – base decisions on classic axiomatic utility theory in order to act rationally

(0/1) Table-driven/reflex agent architecture
(0) Table-driven agents

- **Table lookup** of percept-action pairs mapping from every possible perceived state to optimal action for that state
- **Problems**
  - Too big to generate and to store (Chess has about $10^{120}$ states, for example)
  - No knowledge of non-perceptual parts of the current state
  - Not adaptive to changes in the environment; requires entire table to be updated if changes occur
  - Looping: Can’t make actions conditional on previous actions/states

(1) Simple reflex agents

- **Rule-based reasoning** to map percepts to optimal action; each rule handles a collection of perceived states
- Sometimes called reactive agents
- **Problems**
  - Still usually too big to generate and to store
  - Still no knowledge of non-perceptual parts of state
  - Still not adaptive to changes in the environment; requires collection of rules be updated if changes occur
  - Still can’t make actions conditional on previous state
  - Can be difficult to engineer if the number of rules is large due to conflicts

(2) Architecture for an agent with memory

- Encode “internal state” of the world to remember the past as contained in earlier percepts.
- Note: sensors don’t usually give the entire world state at each input, so environment perception is *captured over time*. “State” is used to encode different “world states” that generate the same immediate percept.
- Requires ability to *represent change* in the world; one possibility is to represent just the latest state, but then can’t reason about hypothetical courses of action.

(2) Agents with memory

- Encode “internal state” of the world to remember the past as contained in earlier percepts.
- Note: sensors don’t usually give the entire world state at each input, so environment perception is *captured over time*. “State” is used to encode different “world states” that generate the same immediate percept.
- Requires ability to *represent change* in the world; one possibility is to represent just the latest state, but then can’t reason about hypothetical courses of action.
(2) Brooks’s Subsumption Architecture

- Rod Brooks, director of MIT AI Lab
- Main idea: build complex, intelligent robots by decomposing behaviors into hierarchy of skills, each completely defining a complete percept-action cycle for one very specific task.
- Examples: avoiding contact, wandering, exploring, recognizing doorways, etc.
- Each behavior modeled by a finite-state machine with a few states (the each may correspond to complex function/module).
- Behaviors are loosely coupled, asynchronous interactions.

(3) Goal-based agents

- **Deliberative** instead of reactive
- Choose actions so as to achieve a goal (given or computed)
- A goal is a description of a desirable situation
- Keeping track of the current state is often not enough - need to add goals to decide which situations are good
- Achieving a goal may require a long action sequence
  - Must model action consequences: "what will happen if I do...?"
  - Planning

(4) A complete utility-based agent
**4) Utility-based agents**

- When there are multiple possible alternatives, how to decide which one is best?
- Goals specify a crude distinction between a happy and unhappy states, but often need a performance measure that describes “degree of happiness.”
- Utility function $U: \text{State} \rightarrow \text{Reals}$ indicating a measure of success or happiness for a given state
- Allows decisions comparing choice between conflicting goals, and choice between likelihood of success and importance of goal (if achievement is uncertain).

---

**Properties of Environments**

- **Fully/Partially observable**
  - If an agent’s sensors give it access to the complete state of the environment needed to choose an action, the environment is **fully observable**.
  - Such environments are convenient, freeing agents from keeping track of the environment’s changes.
- **Deterministic/Stochastic**
  - An environment is **deterministic** if the next state of the environment is completely determined by the current state and the agent’s action; in a stochastic environment, there are multiple, unpredictable outcomes
- In a fully observable, deterministic environment, agents need not deal with uncertainty

---

**Properties of Environments II**

- **Episodic/Sequential**
  - An *episodic* environment means that subsequent episodes do not depend on what actions occurred in previous episodes.
  - In a *sequential* environment, the agent engages in a series of connected episodes.
  - Such environments don’t require the agent to plan ahead.
- **Static/Dynamic**
  - A *static* environment doesn’t change as the agent is thinking
  - The passage of time as an agent deliberates is irrelevant
  - The agent needn’t observe the world during deliberation

---

**Properties of Environments III**

- **Discrete/Continuous**
  - If the number of distinct percepts and actions is limited, the environment is *discrete*, otherwise it is *continuous*.
- **Single agent/Multiagent**
  - If the environment contains other intelligent agents, the agent must be concerned about strategic, game-theoretic aspects of the environment (for either cooperative or competitive agents)
  - Most engineering environments don’t have multiagent properties, whereas most social and economic systems get their complexity from the interactions of (more or less) rational agents.
### Characteristics of environments

<table>
<thead>
<tr>
<th>Characteristics of environments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solitaire</td>
</tr>
<tr>
<td>Backgammon</td>
</tr>
<tr>
<td>Taxi driving</td>
</tr>
<tr>
<td>Internet shopping</td>
</tr>
<tr>
<td>Medical diagnosis</td>
</tr>
</tbody>
</table>

### Characteristics of environments

<table>
<thead>
<tr>
<th>Characteristics of environments</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td>Solitaire</td>
</tr>
<tr>
<td>Backgammon</td>
</tr>
<tr>
<td>Taxi driving</td>
</tr>
<tr>
<td>Internet shopping</td>
</tr>
<tr>
<td>Medical diagnosis</td>
</tr>
</tbody>
</table>
### Characteristics of environments

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Solitaire</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Backgammon</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Taxi driving</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Internet shopping</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Medical diagnosis</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

→ Lots of real-world domains fall into the hardest case!

### Summary

- An agent perceives and acts in an environment, has an architecture, and is implemented by an agent program.
- An ideal agent always chooses the action which maximizes its expected performance, given its percept sequence so far.
- An autonomous agent uses its own experience rather than built-in knowledge of the environment by the designer.
- An agent program maps percepts to actions and updates its internal state.
  - Reflex agents respond immediately to percepts.
  - Goal-based agents act in order to achieve their goal(s).
  - Utility-based agents maximize their own utility function.
- Representing knowledge is important for good agent design.
- The most challenging environments are partially observable, stochastic, sequential, dynamic, and continuous, and contain multiple intelligent agents.