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**
- General properties
  - Reactive to the environment
  - Pro-active or goal-directed
  - Interacts with other agents through communication or via the environment
  - Autonomous
sensors/percepts and effectors/actions?

Humans have

- **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
Example: autonomous taxi

- **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!**
Rationality

• Ideal **rational agents** should, for each percept sequence, act to maximize expected performance measure based on
  (1) the percept sequence, and
  (2) its built-in and acquired knowledge

• Rationality includes information gathering -- 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 extent that its behavior is determined by its experience
• A system isn’t autonomous if 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 percept sequence/action table 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 used to keep track of past states of the world

(3) Agents with goals
   – Agents with a state and goal information describing 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

Use percept sequence/action table to find the next action. Implemented by a (large) lookup table.
(0) Table-driven agents

**Table lookup** of percept-action pairs mapping from every possible perceived state to optimal action for it

**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; entire table must be updated if changes occur

– Looping: Can’t make actions conditional on previous actions/states
(1) Simple reflex agents

- **Rule-based reasoning** maps percepts to optimal action; each rule handles collection of perceived states (aka 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 environment; collection of rules must be updated if changes occur
  - Still can’t condition actions on previous state
  - Difficult to engineer if the number of rules is large due to conflicts
(2) Architecture for an agent with memory

**internal state** used to keep track of past states of the world
(2) Agents with memory

• Encode *internal state* of world to remember past as contained in earlier percepts
  – Note: sensors don’t usually give entire world state at each input, so environment perception is *captured over time*.
  – *State* used to encode different "world states" that generate the same immediate percept

• Requires *representing change* in the world
  – We might represent just latest state, but then can’t reason about hypothetical courses of action
(3) Architecture for goal-based agent

state and **goal information** describe desirable situations allowing agent to take future events into consideration
(3) Goal-based agents

- **Deliberative** instead of reactive
- Choose actions to achieve a goal
- Goal is a description of a desirable situation
- Keeping track of current state often not enough: must add goals to decide which situations are good
- Achieving goal may require long action sequence
  - Model action consequences: \textit{“what happens if I do...?”}
  - Use \textit{planning} algorithms to produce action sequences
(4) a complete utility-based agent

base decisions on utility theory in order to act rationally
(4) Utility-based agents

- For multiple possible alternatives, how to decide which is best?
- Goals give a crude distinction between happy and unhappy states, but often need a performance measure for degree
- Utility function $U: \text{State} \rightarrow \text{Reals}$ gives measure of success/happiness for given state
- Allows decisions comparing choices between conflicting goals and likelihood of success and importance of goal (if achievement uncertain)
Properties of Environments

• Fully/Partially observable
  – If agent’s sensors give complete state of environment needed to choose action, environment is **fully observable**
  – Such environments are convenient, freeing agents from keeping track of the environment’s changes

• Deterministic/Stochastic
  – Environment is **deterministic** if next state is completely determined by current state and agent’s action
  – **Stochastic** (i.e., non-deterministic) environments have multiple, unpredictable outcomes

• In fully observable, deterministic environments agents need not deal with uncertainty
Properties of Environments

• **Episodic/Sequential**
  – In *episodic* environments subsequent episodes don’t depend on actions in previous episodes
  – In *sequential* environments agent engages in a series of connected episodes
  – Episodic environments don’t require agent to plan ahead

• **Static/Dynamic**
  – *Static* environments doesn’t change as agent is thinking
  – The passage of time as agent deliberates is irrelevant
  – The agent needn’t observe world during deliberation
Properties of Environments III

• Discrete/Continuous
  – If number of distinct percepts and actions is limited, environment is discrete, otherwise it’s continuous

• Single agent/Multiagent
  – In environments with other agents, agent must consider strategic, game-theoretic aspects of 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 interactions of (more or less) rational agents
## 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></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Backgammon</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Taxi driving</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Internet shopping</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Medical diagnosis</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

A **Yes** in a cell means that aspect is simpler; a **No** more complex
## Characteristics of environments

<table>
<thead>
<tr>
<th>Environments</th>
<th>Fully observable</th>
<th>Deterministic</th>
<th>Episodic</th>
<th>Static</th>
<th>Discrete?</th>
<th>Single agent?</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></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Taxi driving</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Internet shopping</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Medical diagnosis</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

A **Yes** in a cell means that aspect is simpler; a **No** more complex
## 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></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Internet shopping</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Medical diagnosis</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

A **Yes** in a cell means that aspect is simpler; a **No** more complex
### 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></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Medical diagnosis</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

A **Yes** in a cell means that aspect is simpler; a **No** more complex.
# 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>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Medical diagnosis</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

A **Yes** in a cell means that aspect is simpler; a **No** more complex.
### Characteristics of environments

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

<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>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Medical diagnosis</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

A **Yes** in a cell means that aspect is simpler; a **No** more complex
Summary

• **Agents** perceive and act in an environment, have an architecture and are implemented by an agent program

• **Ideal agents** chooses actions to maximize their expected performance, given percept sequence so far

• **Autonomous agents** use own experience rather than built-in knowledge of environment by designer
Summary

• **Agent programs** map percepts to actions and update their internal state
  – **Reflex** agents respond immediately to percepts
  – **Goal-based** agents act to achieve their goal(s)
  – **Utility-based** agents maximize their utility function

• **Representing knowledge** is important for good agent design

• Most challenging environments are partially observable, stochastic, sequential, dynamic, and continuous and contain multiple agents
Summary

• Not all AI problems a good fit for or require an agent model, e.g., playing solitaire

• Nor are many AI tasks you might be asked to solve:
  – Classify movie reviews as negative, neutral or positive
  – Locate faces of people in an image
  – An efficient theorem prover
  – Learn preferred thermostat settings for each hour of each day of a week