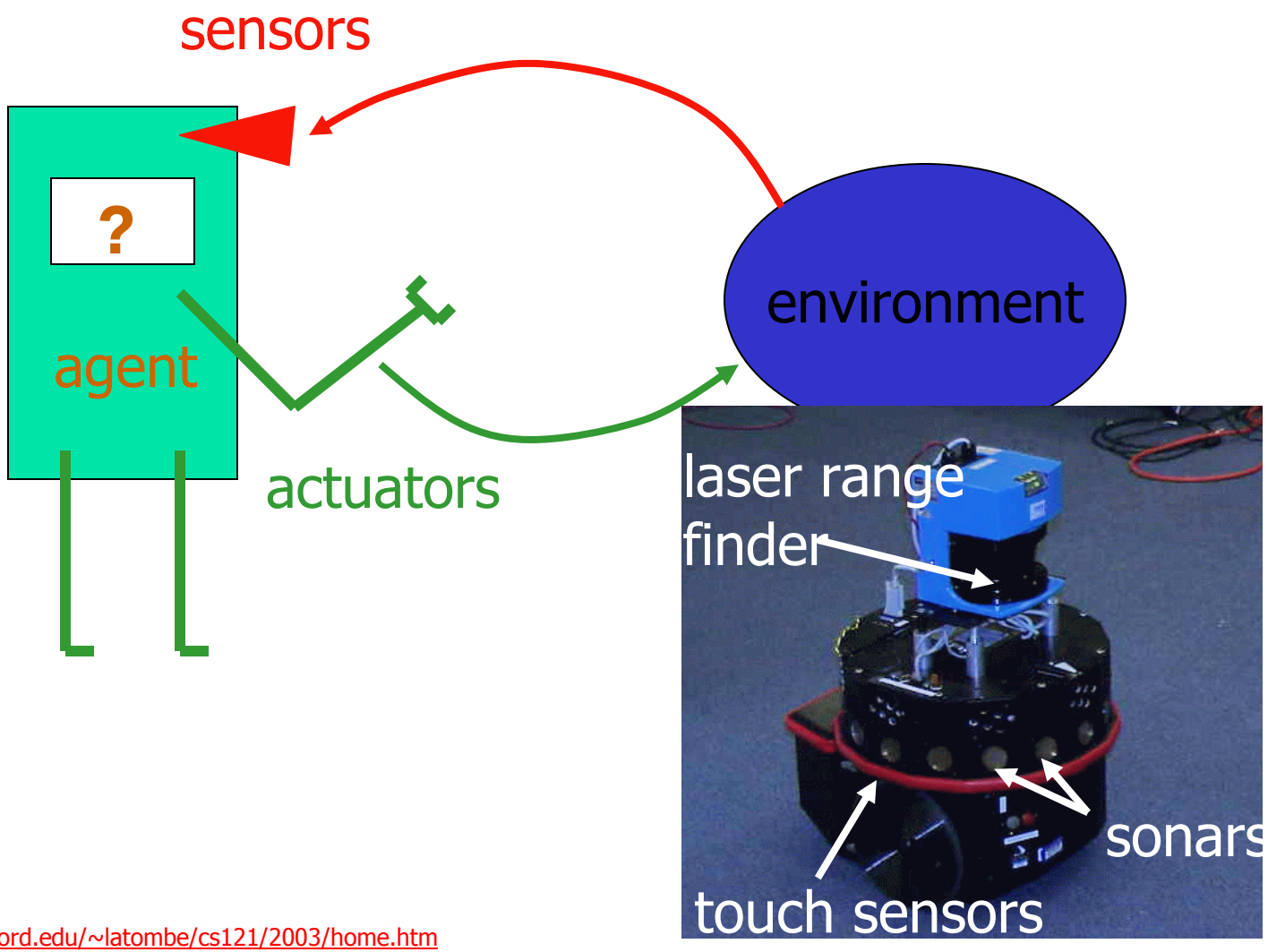
RN, Chapter 2

# Intelligent Agents

- Rational Agent
- PEAS
- Types of Agents

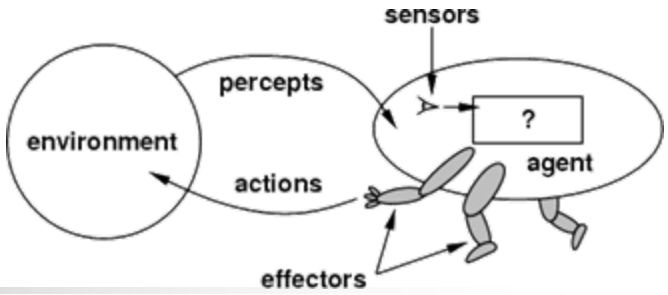
Some material from: D Lin, J You, JC Latombe

# Notion of an Agent




Source: robotics.stanford.edu/~latombe/cs121/2003/home.htm

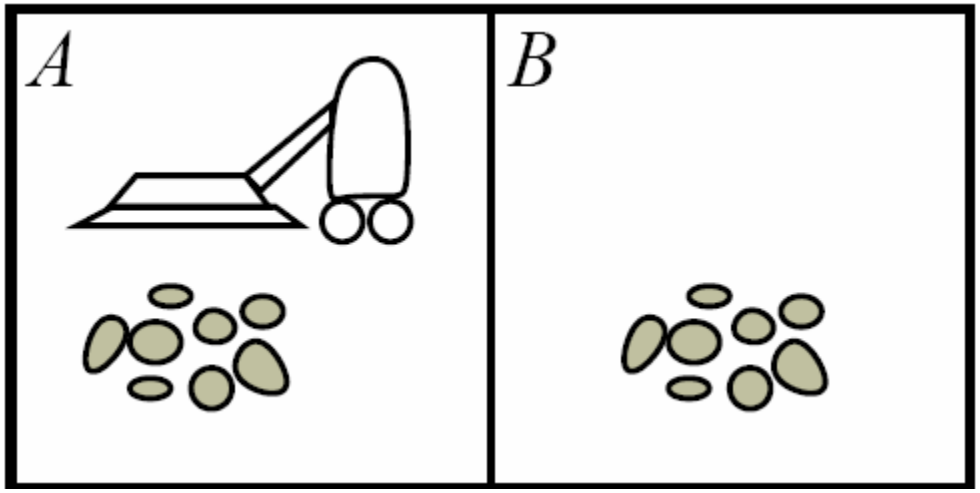
# Rational Agents




- Def'n: A Rational Agent
  - perceives its environment via sensors and
  - acts rationally upon that environment with its actuators

- Agent receives percepts, one at a time, and maps this percept sequence to an action
- Agent specified by **agent function**
  - mapping *percept sequences* to *actions*
- Agent PROGRAM
  - Concisely implements "rational agent function"

# Vacuum-cleaner world

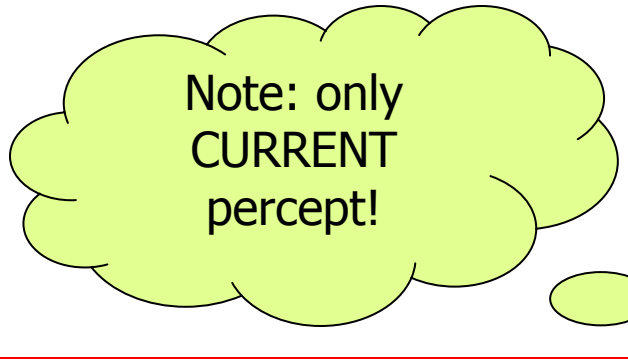



- Percepts: location and contents
  - e.g., [A,Dirty]
- Actions: *Left, Right, Suck,*

# A Vacuum-Cleaner Function

Is this correct function?
… subroutine?

| ***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 |
| … | … |




function Reflex-Vacuum-Agent( [*location,status*]) returns an action
    if *status* = *Dirty* then return *Suck*
    else if *location* = A then return *Right*
    else if *location* = B then return *Left*

# Rationality

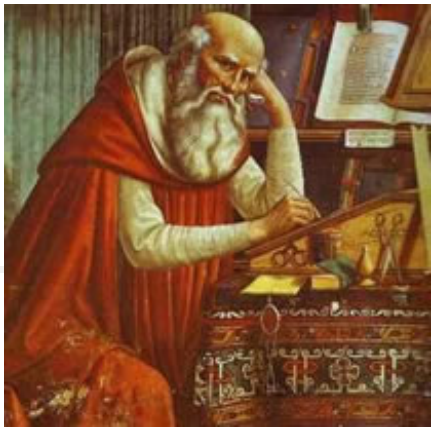
- Fixed performance measure evaluates percept sequence
  - one point per square cleaned up in time T?
  - one point per clean square per time step, minus 1/3 per move?
  - penalize for > k dirty squares?

- Rational action
  - maximizes expected value of performance measure
  - given the *percept sequence to date + prior built-in knowledge*

- So as percept does NOT include entire house, not irrational to act based on single square!
- If sucking is stochastic ( succeed), this

Good (rational) decision, bad outcome

# Rationality…



- Rationality is wrt
  - Imperfect knowledge
  - Limited (computational) resource

Rationality ≠ Omniscient

Rationality ≠ Clairvoyant

Rationality ≠ Successful

- Ignores consciousness, emotions, …
- Doesn't necessarily have anything to do with how *humans* solve the problems

# Environments

Agent design depends on
- performance measure
- type of environment

- **(Fully) Observable**: Percepts determine state
  - ⇒ policy based on current percept
  - ⇒ no need for (internal) state
- **Deterministic**: Actions cause deterministic changes.
  - ⇒ Simpler model of action effects

  (If env. also fully observable, everything is certain)
- **Episodic**: Performance measured on short sequence of actions (then start over)
  - ⇒ No long-term issues
- **Static**: World doesn't change while agent is thinking
  - ⇒ Allows lots of computing per decision
- **Discrete**: Percepts, actions are discrete
  - ⇒ Easier to interpret percepts & to evaluate actions
- **Single Agent vs Multiple**
  - ⇒ Volition on part of external world

# Examples of Environments


| | Crossword Puzzle | Backgammon | Internet shopping | Taxi |
|---|---|---|---|---|
| Observable? | | | | |
| Deterministic? | | | | |
| Episodic? | | | | |
| Static? | | | | |
| Discrete? | | | | |
| Single-agent? | | | | |

… should design agent based on type of environment…

# Types of Agents

- Simple reflex agents
  - Actions determined by sensory input only
- Model-based reflex agents
  - Has internal states
- Goal-based agents
  - Action may be driven by a goal
- Utility-based agents
  - Maximizes a utility function

Search!

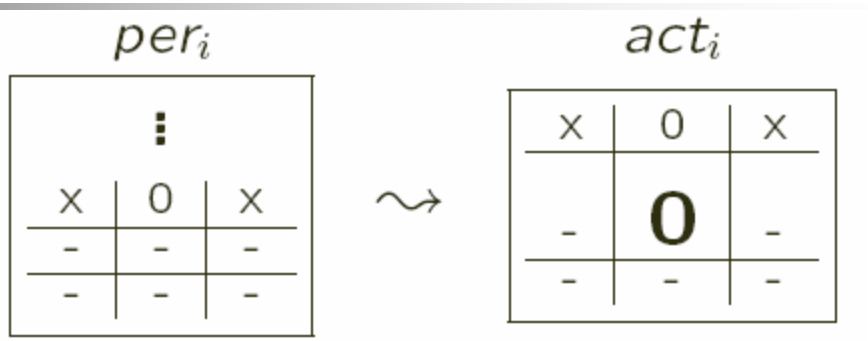
Logic!

Planning!

Decision Theory!

# Implementing Reflex Agents using Tables

- TicTacToe:




- Find percept $per_i$ in percept-lookup-table, apply associated action $act_i$

PROBLEMS:

- Requires keeping EVERY possible sequence of perceptions:
  - Chess: $35^{100}$ entries!
- Requires DESIGNER to design appropriate response to every sequence

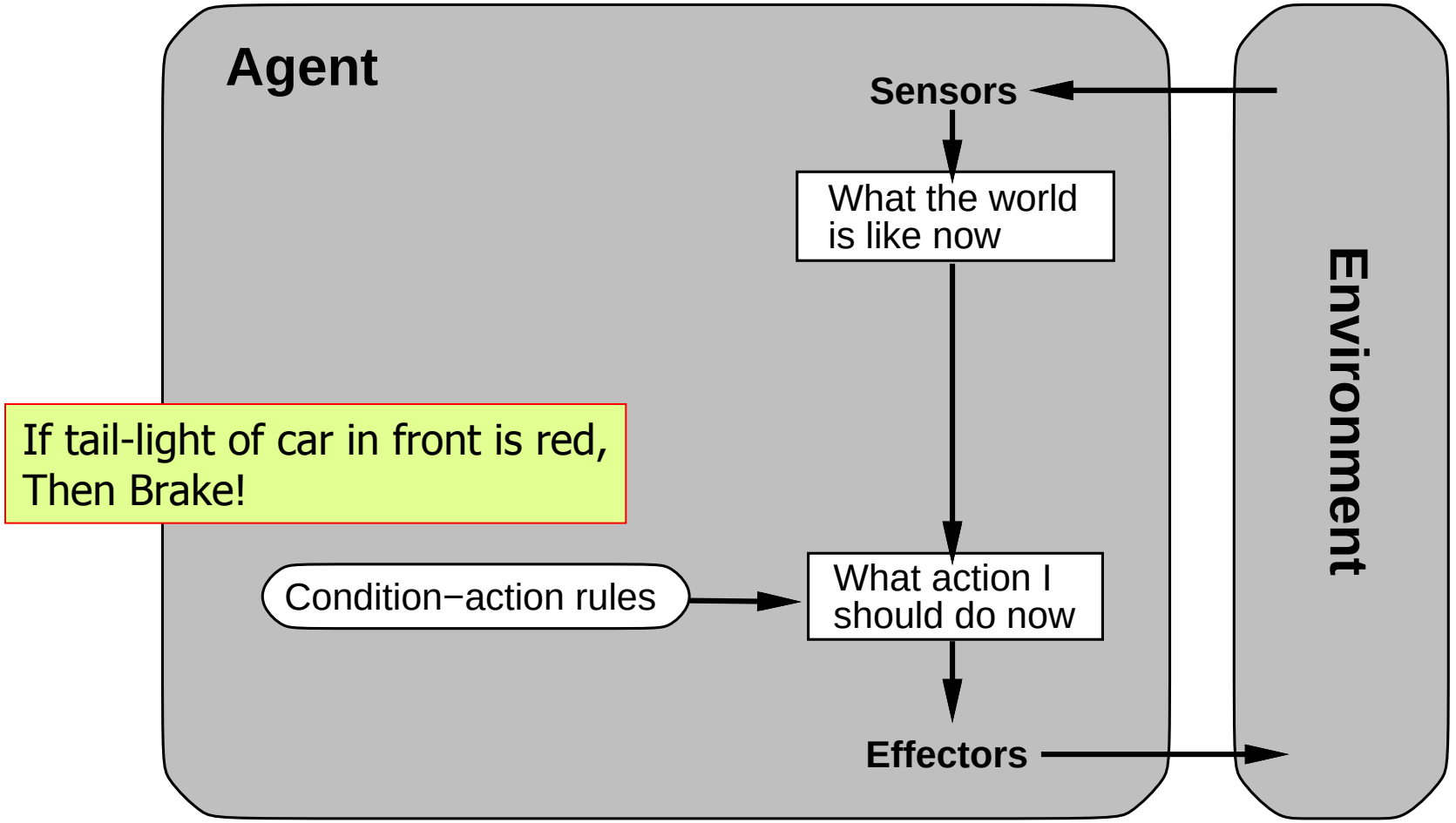
# Avoiding Intractably Large Tables

- **Compact Representations of Table:**
  Many actions in table are identical !
  - **Markov Environments**:
    Eg: In Chess, action depends on only current board position;
    independent of all previous percepts…
  - **Irrelevant Percepts:**
    Eg: If car in front of you slows down, you should brake;

    independent of… { color/model of car; music on radio; temperature; … }

- Use **Condition-Action rules**, with
  - Condition specifying ONLY relevant information
  - of CURRENT percept

# Simple Reflex Agent (rules)

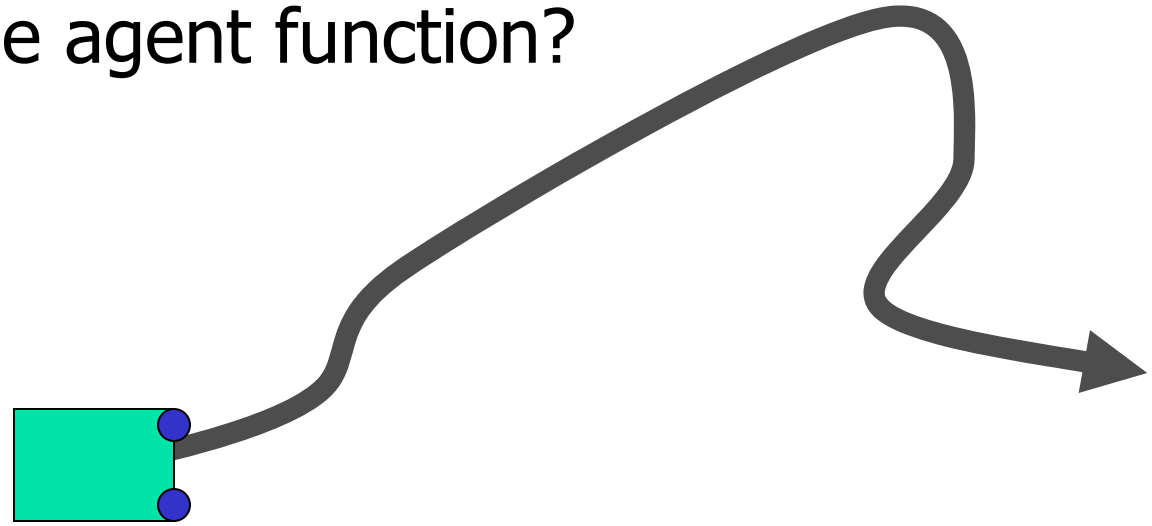


# Example

- A LEGO MindStorm™ program:

```
if (isDark(leftLightSensor))
   turnLeft()
else if (isDark(rightLightSensor))
  turnRight()
else goStraight()
```

- What's the agent function?



# Summarizing Percept Sequence

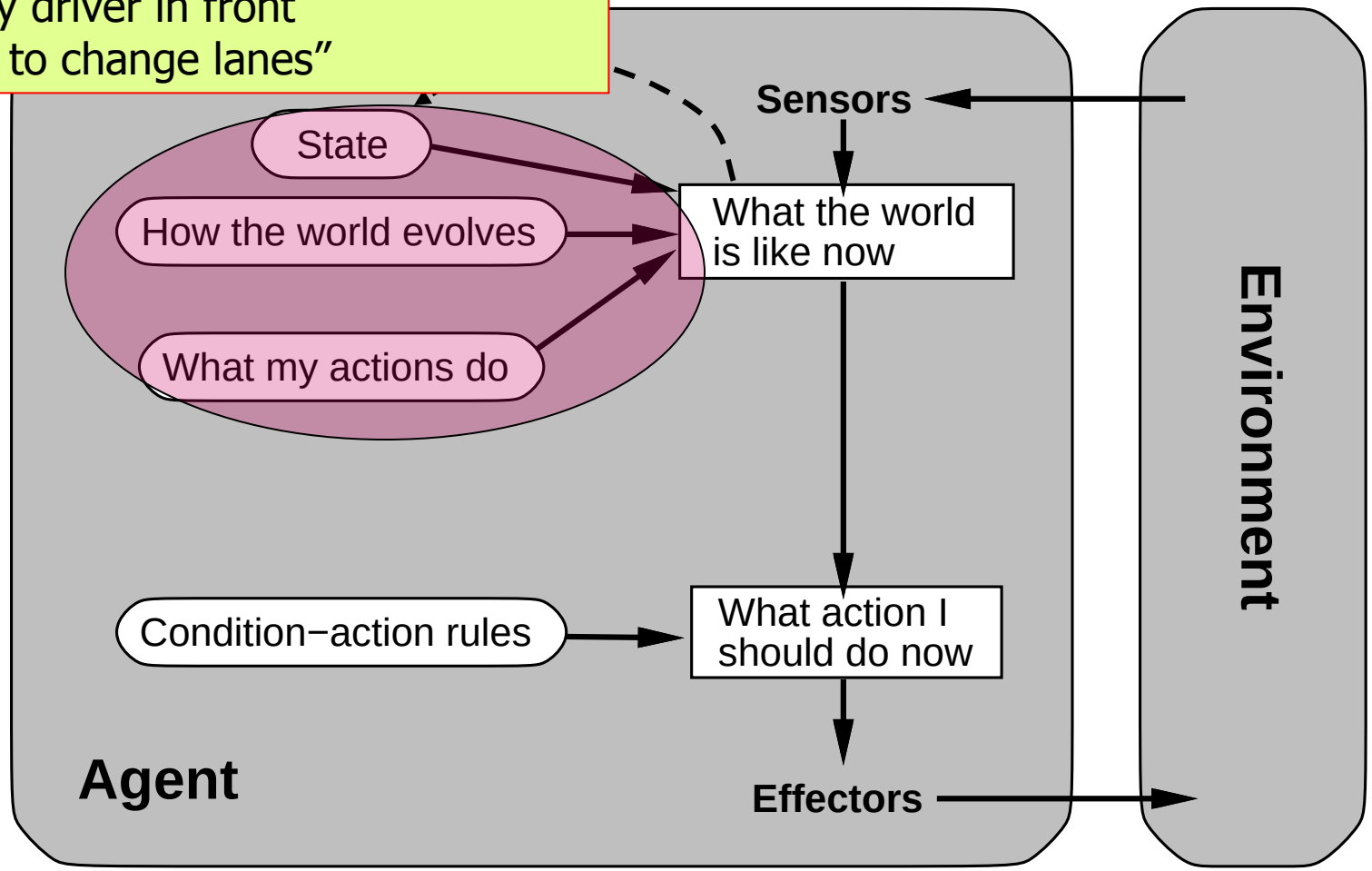
- May require percepts *over time* to understand situation

$\Rightarrow$ Use "state" $\approx$ synopsis of percept history

- Eg: To identify "Crazy Driver"
  - requires sequence of percepts
  - summarize as state= "Behind Crazy Driver"
- Make decision based on
  - STATE, as well as
  - current percept

# Model-Based Reflex Agent

Deal with Sequence of Observations:
"Crazy driver in front"
"Safe to change lanes"

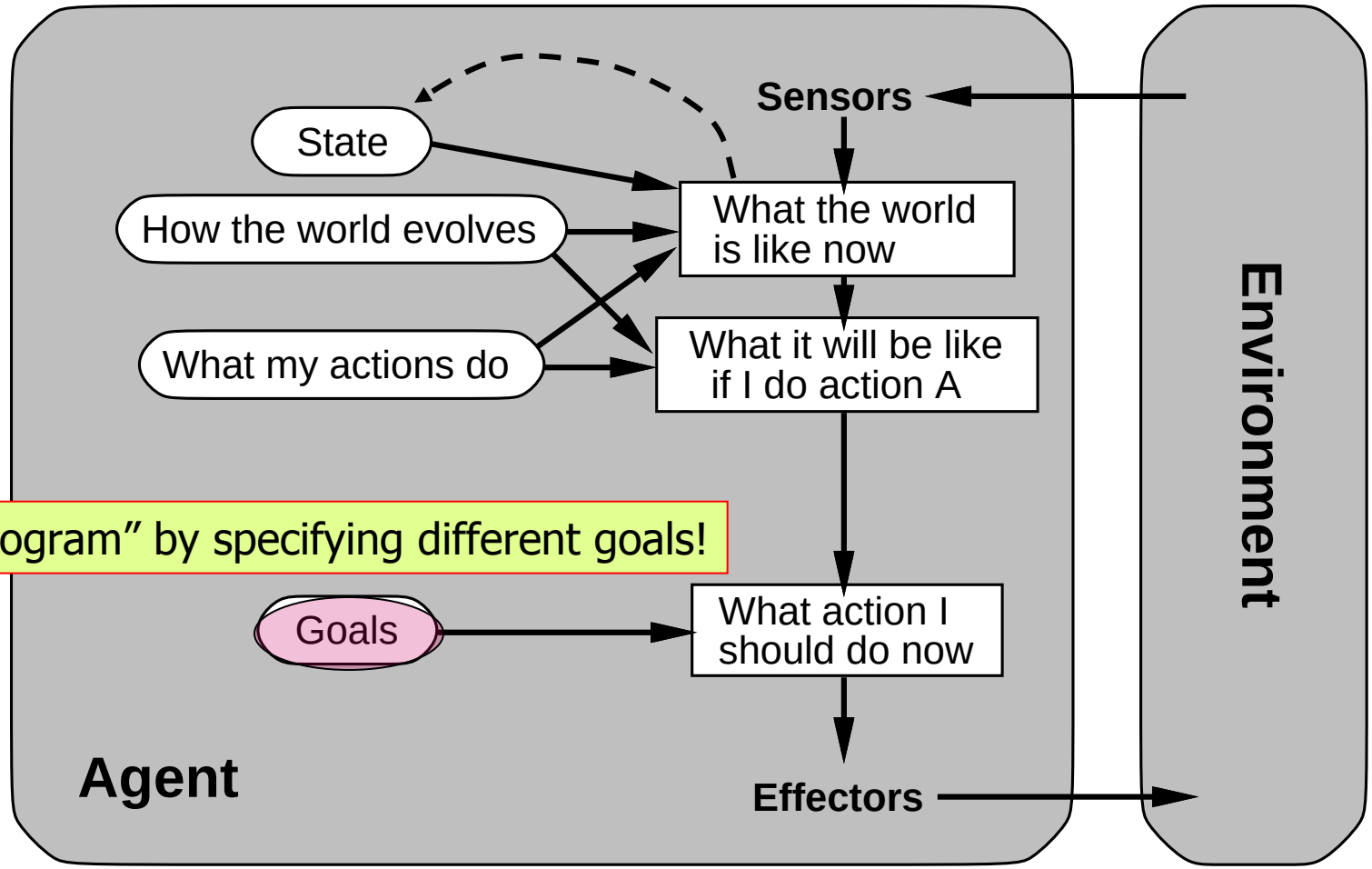


# Problem with Reflex Agent

- Reflex Agent:
  - given state s & percept p
  - performs action action(s,p)
- Reflex Taxi Driver
  - "Take Fox Drive @ Whitemud + Fox"
  - Great if goal is "Get to University"
  - What if goal is "West Edmonton Mall"?
- Want agents that can achieve DIFFERENT goals…
  … short of re-programming

# Goal-based

- Generate possible sequences of actions
- Predict resulting states
- Assess goals in each resulting state
- Choose an action that achieves goal




"Reprogram" by specifying different goals!

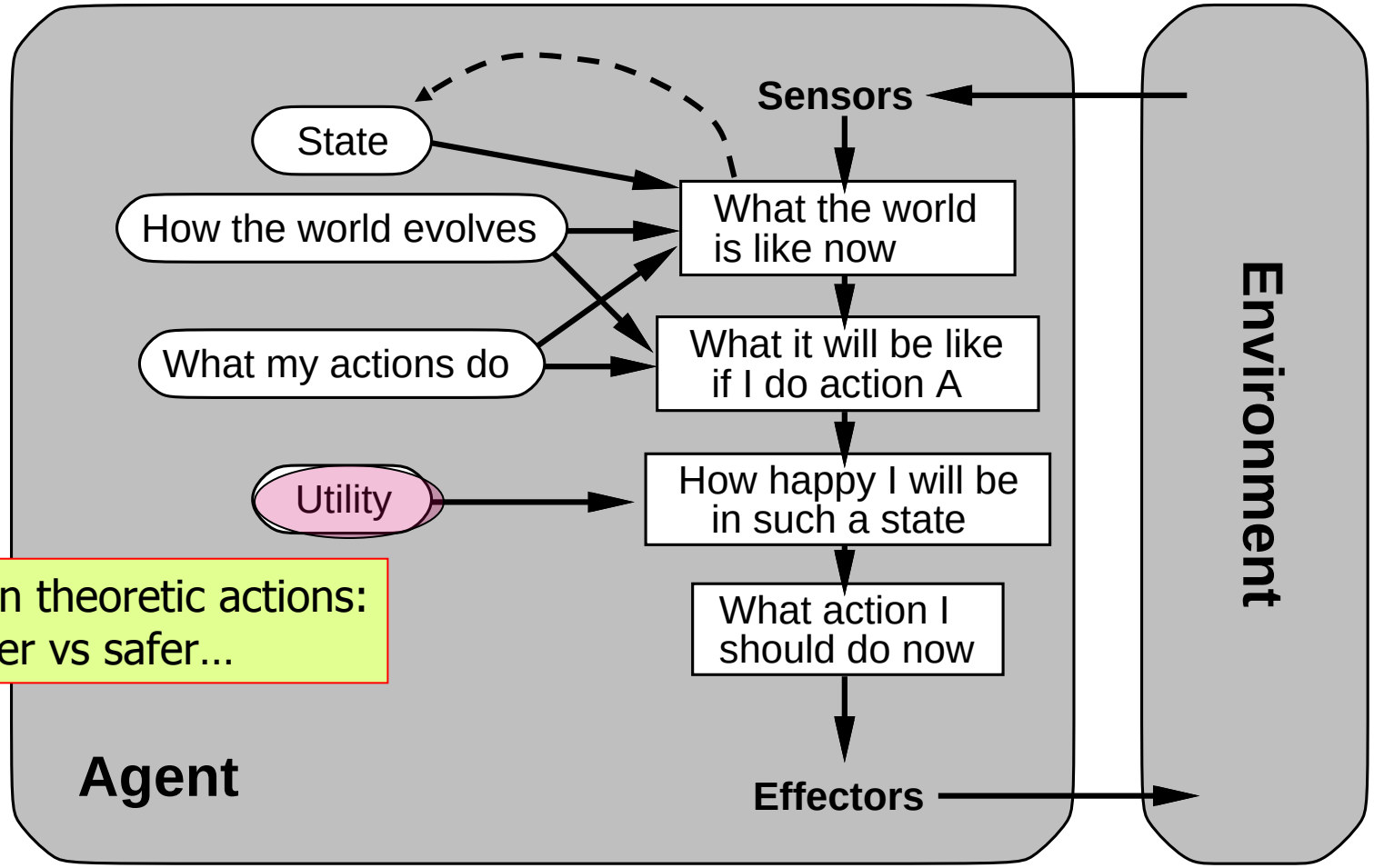
# Utility-Based Agents

- While goals are qualitative,
  (eg, get to destination)
- … may want to make quantitative comparisons of actions
  - (eg, quicker, safer, more reliable paths)

$\Rightarrow$ Utility fn $U: State \rightarrow \mathcal{R}$

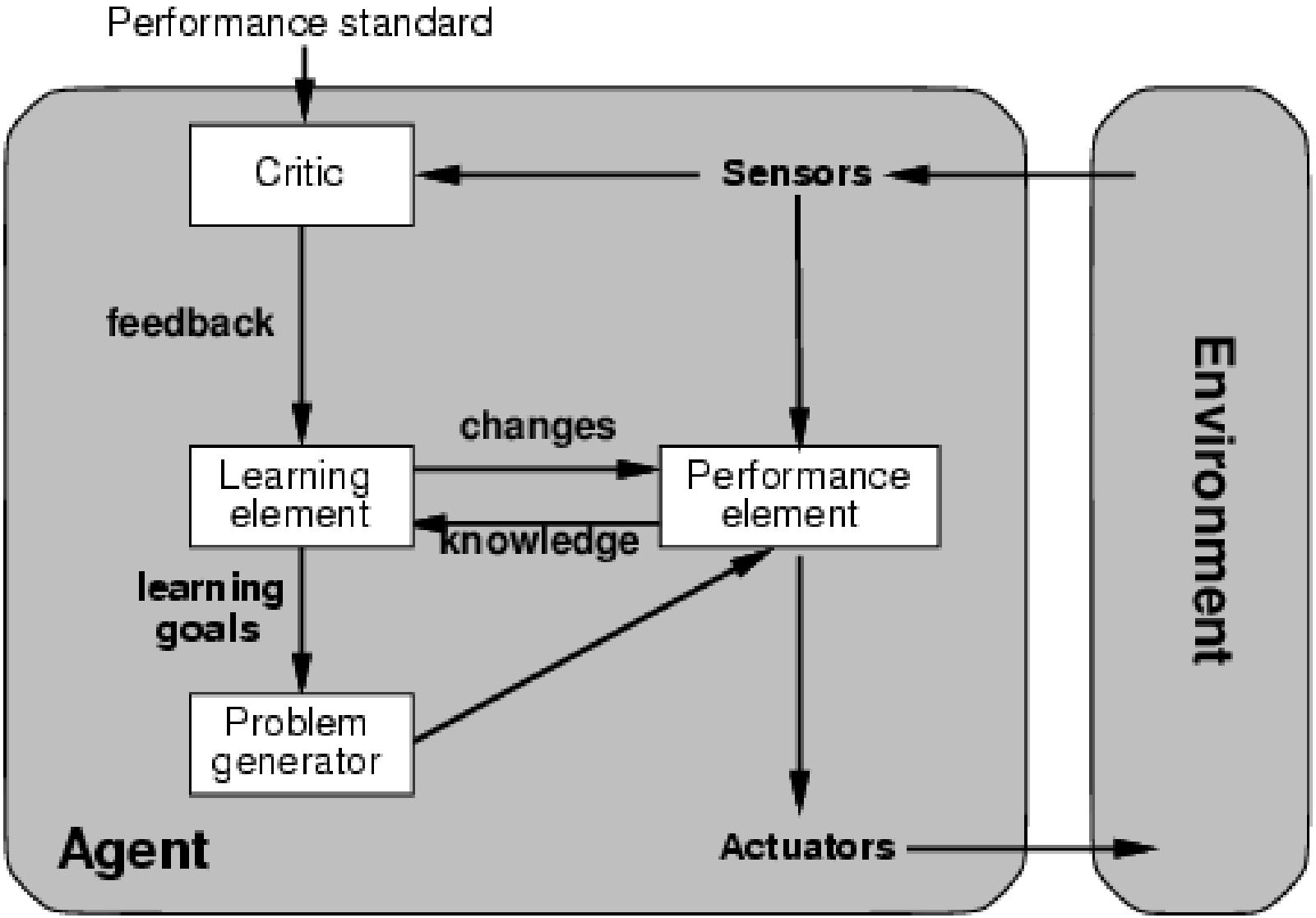
- Important for making tradeoffs
  - speed vs safety
  - likelihood of success against importance of goal
  - …

# Utility-based Agent




Decision theoretic actions:
... faster vs safer...

# Learning agents



# Types of Environments/Agents

- **Search-Based Agents**
  Environment: observable, deterministic, static, discrete
  . . . known, modeled using arbitrary code
- **Logical Agents**
  Environment: observable, deterministic, *, discrete
  . . . known, modeled using logical inference
- **Decision-Theoretical Agents**
  Environment: observable, nondeterministic, *, *
  . . . known, modeled using belief networks
- **Learning Agents**
  Environment: observable, *, *, *
  . . . unknown

# Course Outline

- Introduction
  - What is (isn't) AI? [Ch1]
  - Intelligent Agents [Ch2]
- Search-based Agents
  - Blind (Uninformed) Search [Ch3]
  - Heuristic (Informed) Search [Ch4]
  - Constraint satisfaction [Ch5]
  - Local search, stochastic search (GSAT)
  - Game playing [Ch6]
- Logical Agents
  - Foundations (Reasoning, Entailment, Derivation, . . . ) [Ch7,8]
  - ? Representation (Simulation, Diagnosis) [Ch10]
  - Resolution, Implemented Systems [Ch10]
  - Planning [Ch11]
- Decision-theoretical Agents
  - Intro Probability [Ch13]
  - Bayesian Belief Nets [Ch14]
  - Utility, Influence diagrams [Ch16]
  - Sequential decision-making (MDP, dynamic belief/decision networks) [Ch17]
  - Game Theoretic agents [Ch17.6]
- Learning Agents [Ch18]

# Summary

- Agents interact with environments using actuators and sensors
- *Agent function* describes what agent should do in any situation
- *Performance measure* evaluates environment sequence
- Ideal rational agent maximizes *expected performance*
- Agent programs implement (some) agent functions
- PEAS descriptions define task environments
- Environments are categorized along several dimensions:
  observable? deterministic? episodic? static? discrete? single-agent?
- Several basic agent architectures exist:
  reflex, reflex with state, goal-based, utility-based