AI and Autonomous Agents

Lecture 2

- Did everybody get my e-mail?
- Class cancelled on Feb 15 - Monday evening LAB TIMES

- A Brief History of AI
- Taxonomize Autonomous Agents and their Environments

brief history of AI (1943–56)

- Warren McCulloch & Walter Pitts (1943)

– artificial neural network (ANN), drew on these sources:

basic physiology of neurons in the brain

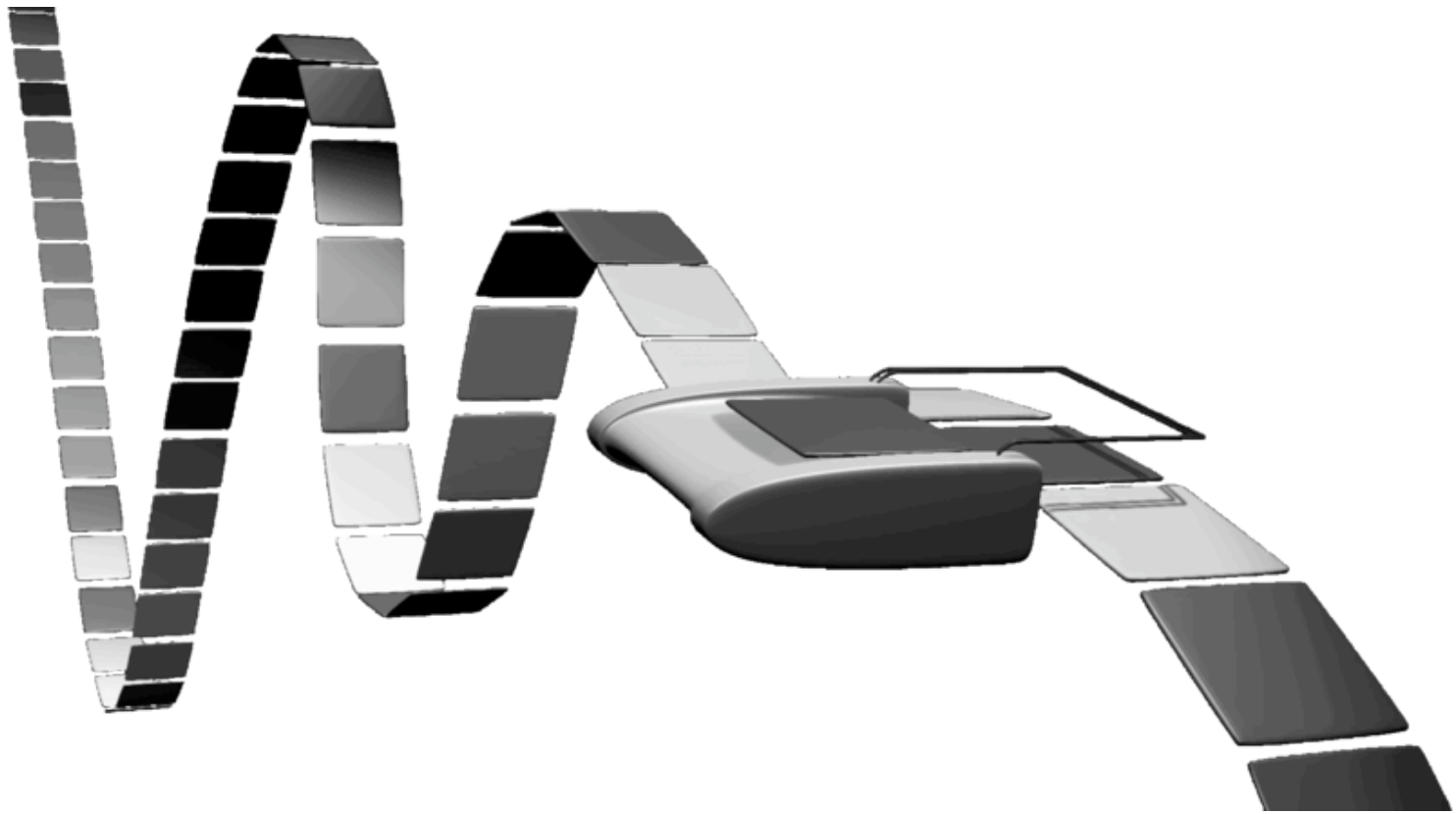
propositional logic

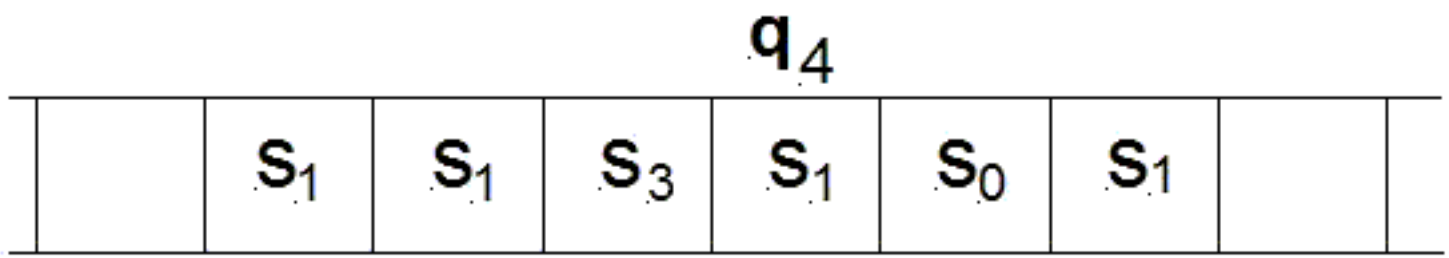
Turing's theory of computation

– neurons are either "on" or "off"; connections between them are used to "compute" functions

– ANN proved equivalent to Turing machine

ANN proved equivalent to Turing machine





Early Enthusiasm, Great Expectations (1952-1969)

- Donald Hebb (1949) — showed that ANN's could "learn" by changing strengths of connections between neurons (*Hebbian Learning*)
- Alan Turing (1950) — "Computing Machinery and Intelligence" - *Set AI's Vision*
- Claude Shannon (1950) — first chess playing program
- Marvin Minsky (1951) — first neural net computer — SNARC

  (3000 Vacuum Tubes, automatic pilot mechanism from B-24 bomber, 40 neurons)
- Dartmouth College (1956) AI WORKSHOP

  term "AI" coined by John McCarthy

  Allen Newell & Herb Simon presented LOGIC THEORIST program

Early Enthusiasm, Great Expectations (1952-1969)

• John McCarthy (1958) — invents "LISP"

• McCarthy vs Minksy:

– McCarthy — representation, reasoning in formal logic

– Minsky — anti-logic; just make programs work! (**microworlds**) — toy problem domains

example: blocks world, general problem solver (GPS)

• programs written that could:

plan, learn, play games, prove theorems, solve problems.

• major centers established:

– Marvin Minsky — MIT

– John McCarthy — Stanford

– Allen Newell & Herb Simon — CMU

"It'll scale, honest. . . "

Early Enthusiasm, Great Expectations (1952-1969)

Herbert Simon (1957):

"It is not my aim to surprise or shock you - but the simplest way I can summarize is to say that there are now in the world machines that think, that learn and that create. Moreover, their ability to do these things is going to increase rapidly until - in the visible future - the range of problems they can handle will be coextensive with the range to which the human mind has been applied."

"visible future" ?

Period of Recession for AI (1970's)

- 1970's (Lighthill report in UK)
- techniques developed on microworlds would not scale
- implications of complexity theory developed in late 1960s, early 1970s began to be appreciated:

– brute force techniques will not work (**intractable**)

– works in principle does not mean works in practice (**lack of generality**)

– problems of basic structures (*Perceptrons* by Minsky and Papert)

Knowledge Based Systems, Industry (1980s)

• general purpose, brute force techniques don't work, so use knowledge rich solutions

• early 1980s saw emergence of **expert systems** as systems capable of exploiting knowledge about tightly focused domains to solve problems normally considered the domain of experts (also called "weak methods")

  • expert systems success stories:

  – DENDRAL — interpreting mass spectrometers, used by chemists

  – MYCIN — blood diseases in humans (added uncertainty factors)

• Ed Feigenbaum's knowledge principle: mapping of general, "first principles" to efficient special forms ("cookbook recipes") (many **if .. then ...** rules)

• rise of natural language processing ("There is no such thing as syntax")

  - General Knowledge about the World and General Method for using that Knowledge

Knowledge Based Systems (1980s)

• **expert systems** emphasized knowledge representation: rules, frames, semantic nets

• problems:

– the knowledge elicitation bottleneck (interviews etc...)

– flexibility with a changing world

– knowing when there is no answer

– lack common sense and human creativity to answers

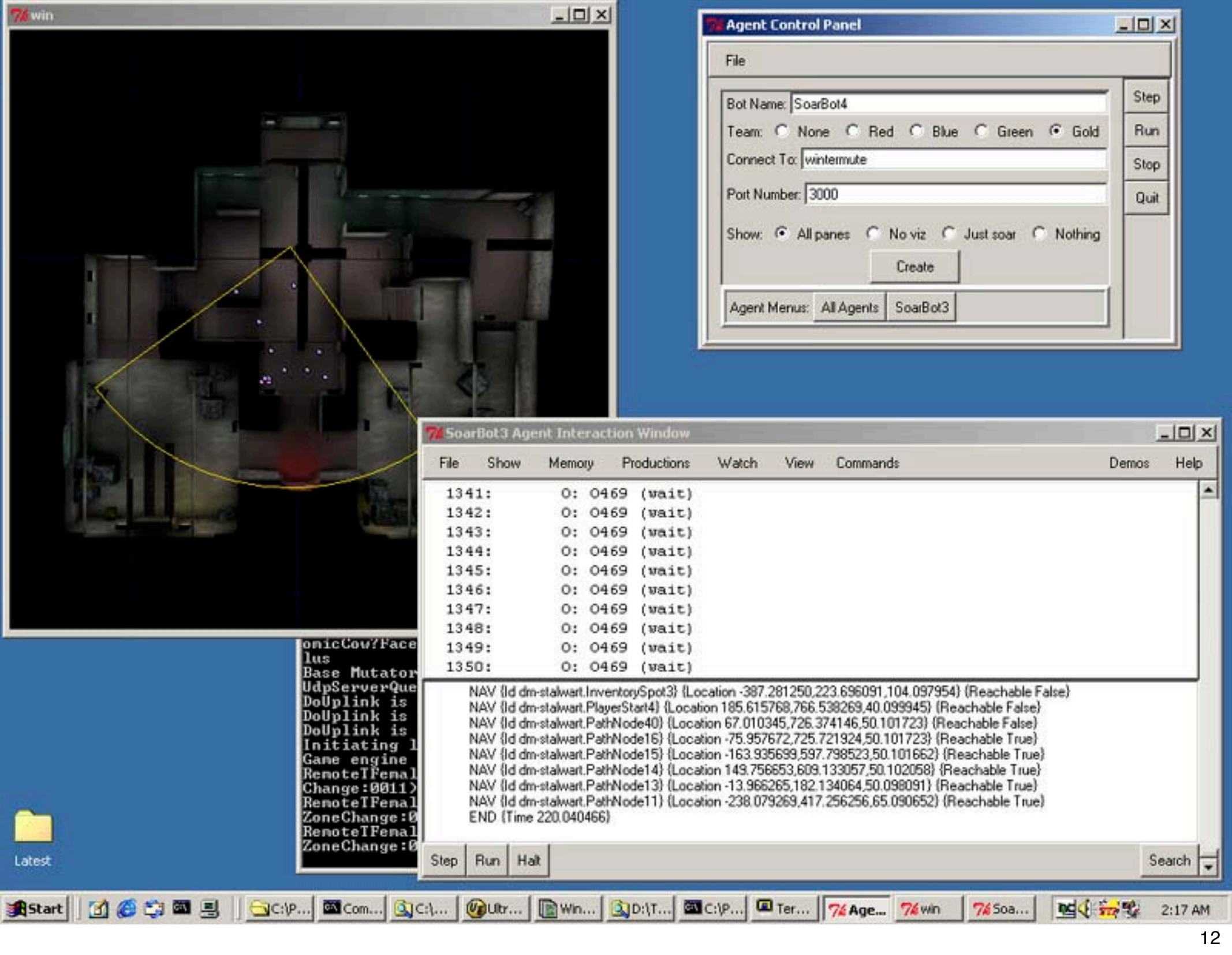
– marrying expert system and traditional software (written PROLOG and LISP)

– breaking into the mainstream

Intelligent Agents (1980s)

- "the AI winter"
- most companies set up to commercialize expert systems technology went bust
- 1990s: emphasis on understanding the interaction between agents and environments
- AI as component, rather than as end in itself
- rise of the "intelligent agent"

  SOARbot (Newell, Laird et al.) - plays *Unreal Tournament*, and Apache Helicopters


win
Agent Control Panel
File
Bot Name: SoarBot4
Team: None Red Blue Green Gold
Connect To: wintermute
Port Number: 3000
Show: All panes No viz Just soar Nothing
Create
Agent Menus: All Agents SoarBot3
Step
Run
Stop
Quit
SoarBot3 Agent Interaction Window
File Show Memory Productions Watch View Commands Demos Help
1341: O: O469 (wait)
1342: O: O469 (wait)
1343: O: O469 (wait)
1344: O: O469 (wait)
1345: O: O469 (wait)
1346: O: O469 (wait)
1347: O: O469 (wait)
1348: O: O469 (wait)
1349: O: O469 (wait)
1350: O: O469 (wait)
NAV (Id dm-stalwart.InventorySpot3) (Location -387.281250,223.696091,104.097954) (Reachable False)
NAV (Id dm-stalwart.PlayerStart4) (Location 185.615768,766.538269,40.099945) (Reachable False)
NAV (Id dm-stalwart.PathNode40) (Location 67.010345,726.374146,50.101723) (Reachable False)
NAV (Id dm-stalwart.PathNode16) (Location -75.957672,725.721924,50.101723) (Reachable True)
NAV (Id dm-stalwart.PathNode15) (Location -163.935699,597.798523,50.101662) (Reachable True)
NAV (Id dm-stalwart.PathNode14) (Location 149.756653,609.133057,50.102058) (Reachable True)
NAV (Id dm-stalwart.PathNode13) (Location -13.966265,182.134064,50.098091) (Reachable True)
NAV (Id dm-stalwart.PathNode11) (Location -238.079269,417.256256,65.090652) (Reachable True)
END (Time 220.040466)
Step Run Halt Search
Latest
Start
2:17 AM

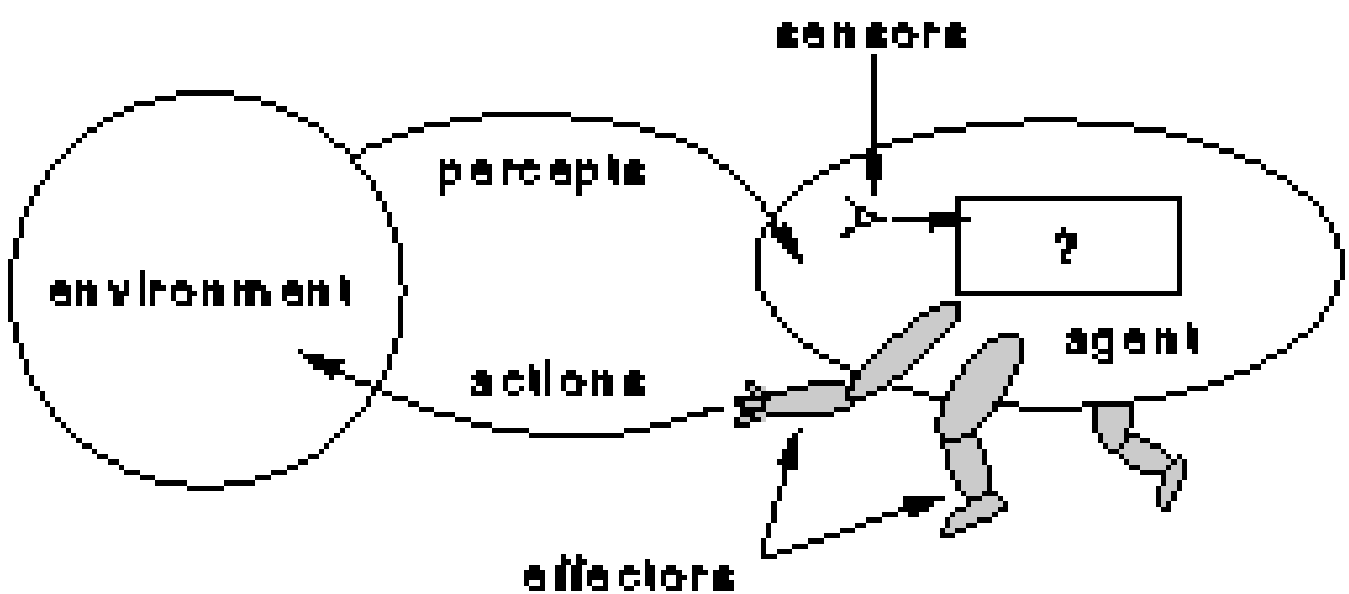
Different ...

... but Complementary Views on AI:

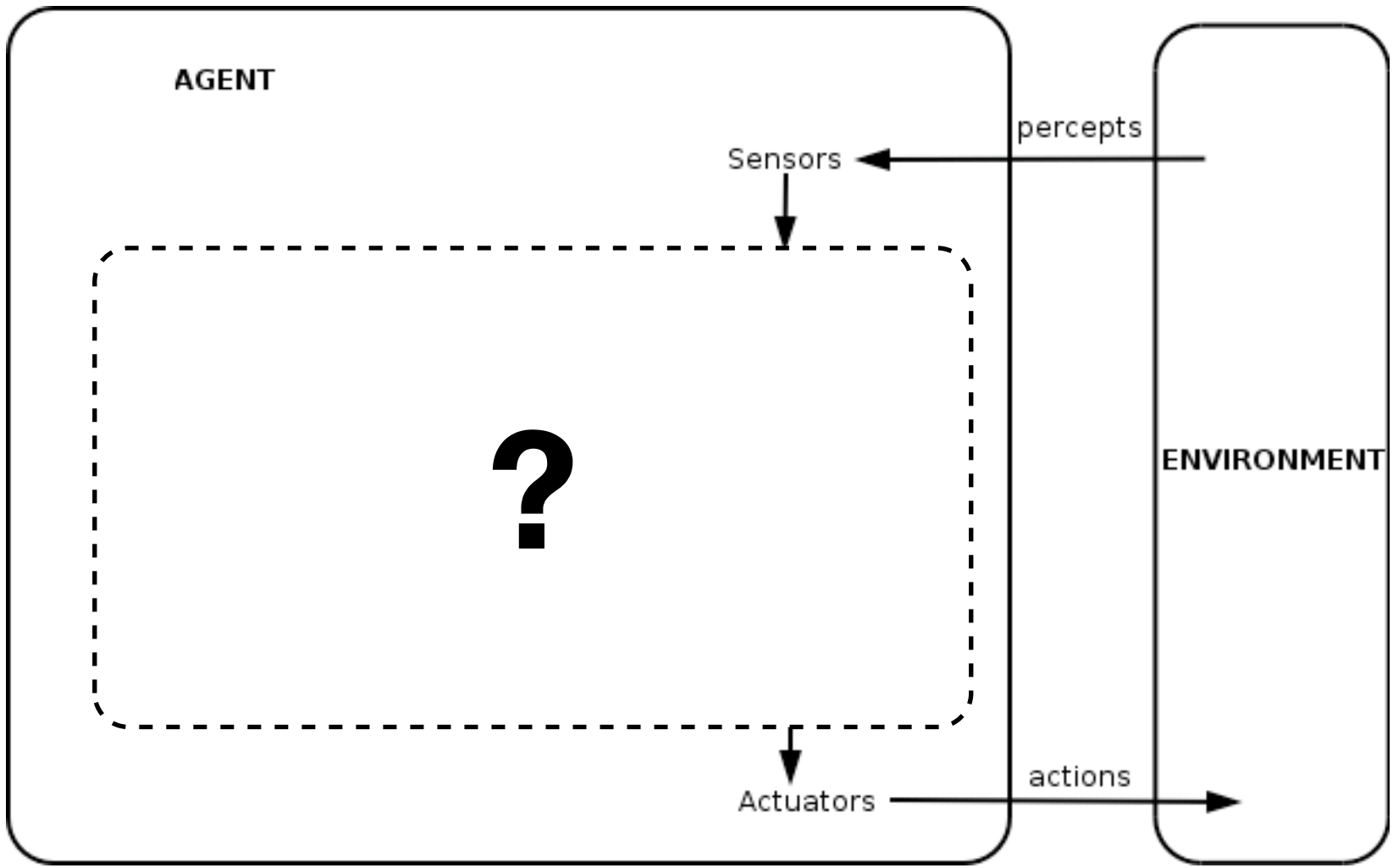
- connectionist: Rumelhart, McClelland
- symbolic: Newell and Simon
- logic: McCarthy

Intelligent Agents

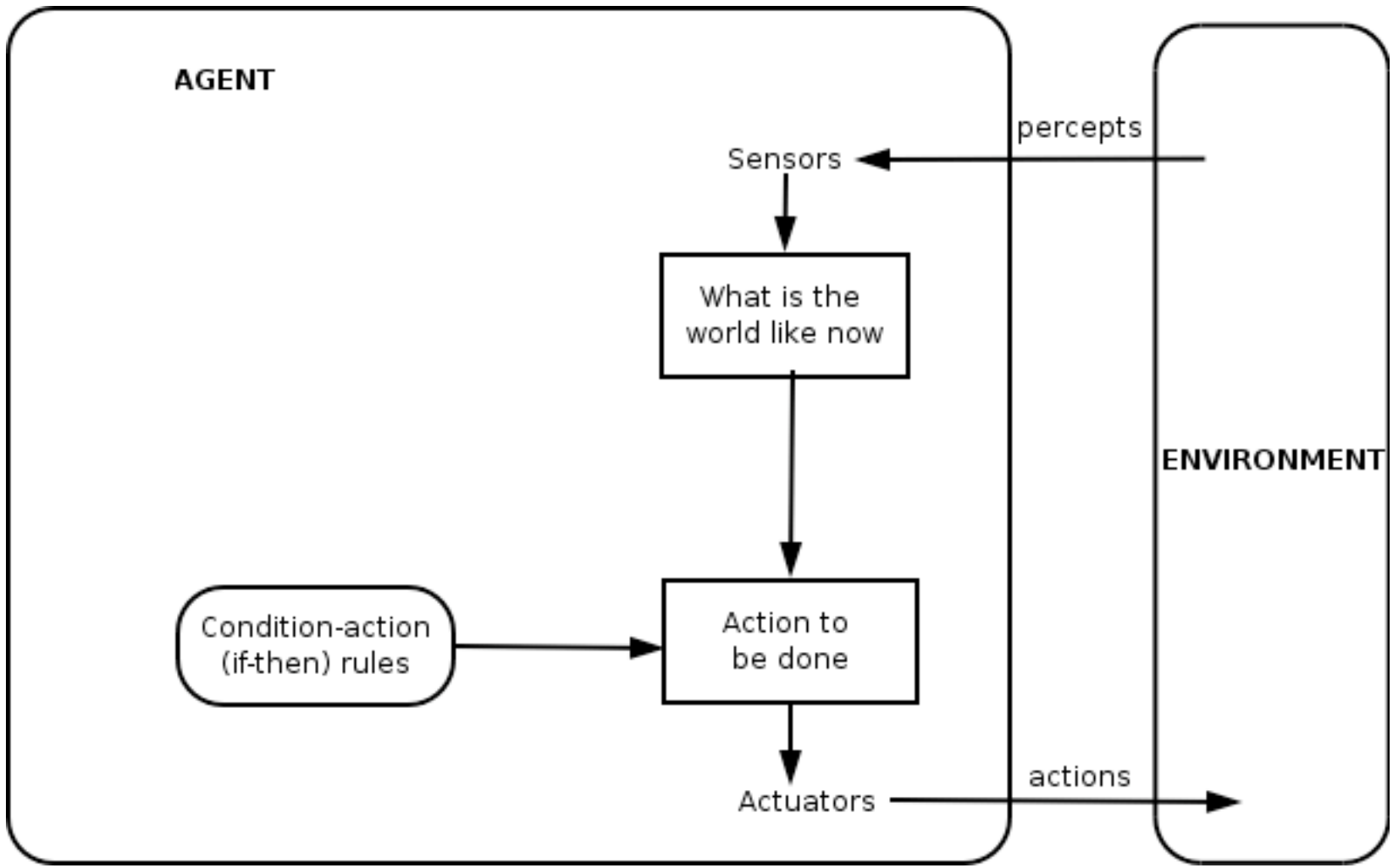
- An **agent** is a system that is situated in an environment, and which is capable of perceiving its environment and acting in it to satisfy its design objectives.



Agent Framework



A Reflex Agent



Examples

Agent Type	Environment	Sensors	Effectors
Human	physical world	eyes, ears...	hands, legs, spoken...
Software	OS (i.e. UNIX)	ls, ps...	rm, chmod...
Internet Agent	Internet (i.e. TCP/IP)	http requests	http commands
Embodied (robotic) agent	physical world (perhaps prepared)	light meters, bumpers, thermometers, . . .	wheels, treads, legs, grippers...

What to do?

Those who do not reason

Perish in the act.

Those who do not act

perish for that reason

(W H Auden)

- The key problem we have is knowing the right thing to do.
- Knowing what to do can in principle be easy:

  consider all the alternatives, and choose the "best".

- But Auden's quote! In any time-constrained domain, we have to make a decision in time for that decision to be useful !
- A tradeoff.

"Deep Blue will be able to explore 200,000,000 positions per second. Incidentally, Garry Kasparov can examine approximately three positions per second." (IBM Research Team)

Performance Measure

Need to know how and when to evaluate success.

- how:
  - an objective performance measure
  - application specific
- when:
  - in discrete episodes, or over long periods? (cleaning the floor over 8 hours)
- don't confuse omniscience with rationality (meteors falling from the sky)
- real agents don't know enough to always make the best choice.

  (We often fall into this trap when making judgments about history.)
- Rationality concerned with expected success given information available.

Ideal rational agent:

- For each **percept sequence**, an ideal rational agent will act to **maximize** its **expected** performance measure, on the basis of information provided by percept sequence plus any information built in to agent.

- (Note that this does not preclude performing actions to find things out. **Information Gathering:**

    a. Looking both ways before crossing the street.

    b. Exploring

- More precisely, we can view an agent as a function:

$$f : P^* \longrightarrow A$$

Mapping from sequences of percepts *P* to actions *A*.

Example: Quadratic Agent

Percept	Action
0	0
1	1
2	4
3	9
4	16
...	...

- This table can be viewed as a specification of the agent.
- We don't have to implement agent as table lookup:

```
int agent(int n)
{
  return n * n;
}
```

Autonomy

- Autonomy a crucial concern for agents.
- Means behavior is based on own experience.
- Implies learning, or adaptation.

Structure of Agents

Two components:

– **program**: the thing which defines the mapping from percept sequences to actions;

– **architecture**: the "shell" into which the agent program fits (device/artifact running the program, sensors, actuators)

Agent = *program* + *architecture*.

• An appropriate architecture can make design of programs much easier. (LEGO Robots)

Classification of Agents

The **PAGE** approach:

– **p**ercepts;

– **a**ctions;

– **g**oals;

– **e**nvironment.

Agent	Percepts	Actions	Goals	Environment
Refinery Controller	Temp, pressure, chemical sensors	open/close valves, switch on/off heaters	maximise purity yield, safety	Refinery, its operators
Medical Diagnosis System	Symptoms, findings, patients answers	questions, test, treatments	healthy patient, minimize costs	patient, hospital
Email Manager	email arrived, headers, content of email headers, con	delete email, sort email, obtain user attention	present important email first, hide junk mail, minimize email tardiness	mail reader, OS, internet

Fully Observable vs. Partially Observable

(aka Accessible vs. Inaccessible)

- An fully observable environment is one in which the agent can obtain complete, accurate, up-to-date (*relevant*) information about the environment's state.
- Most moderately complex environments (including, for example, the everyday physical world and the Internet) are partially observable.
- The more fully observable an environment is, the simpler it is to build agents to operate in it.
- (Fully-Observable Agents do not need Internal state to track the world)

Deterministic v.s Stochastic (aka non-deterministic)

- A deterministic environment is one in which any action has a single guaranteed effect —

  there is no uncertainty about the state that will result from performing an action.

- The physical world can to all intents and purposes be regarded as non-deterministic.
- Non-deterministic environments present greater problems for the agent designer.
- **Strategic** - deterministic except for actions of other agents

Episodic vs Sequential (non-episodic)

- In an episodic environment, the performance of an agent is dependent on a number of discrete episodes, with no link between the performance of an agent in different scenarios.
- An example of an episodic environment would be a mail sorting system (classification problems).
- Episodic environments are simpler from the agent developer's perspective because the agent can decide what action to perform based only on the current episode — it need not reason about the interactions between this and future episodes.
- Example of a Sequential Environment

Static vs Dynamic

- A static environment is one that can be assumed to remain unchanged except by the performance of actions by the agent.
- A dynamic environment is one that has other processes operating on it, and which hence changes in ways beyond the agent's control. The physical world is a highly dynamic environment.
- If the agent's performance score drops with the passage of time, but the environment does not change, then the environment is **semi-dynamic**.

Discrete vs Continuous

- An environment is discrete if there are a fixed, finite number of actions and percepts in it.
- Russell and Norvig give a chess game as an example of a discrete environment, and taxi driving as an example of a continuous one.
- What about Machine Vision?

Single v.s Multiagent

- What constitutes an agent?
  - If one agent's behavior is maximizing a performance measure which is dependent on another object's behavior.
- City-Driving:
  - Competitive: parking
  - Cooperative: avoiding grid-lock

Summary

- This lecture has looked at:
  - The history of AI
  - The notion of intelligent agents
  - A classification of agent environments.
- Broadly speaking, the rest course will cover the major techniques of AI, with special reference to agents.
- The techniques we'll look at will start with those applicable to simple environments and move towards those suitable for more complex environments.