Intelligent Agents

CHAPTER 2

Oliver Schulte

Outline

- Agents and environments
- Rationality
- PEAS (Performance measure, Environment, Actuators, Sensors)
- Environment types
- Agent types

The PEAS Model

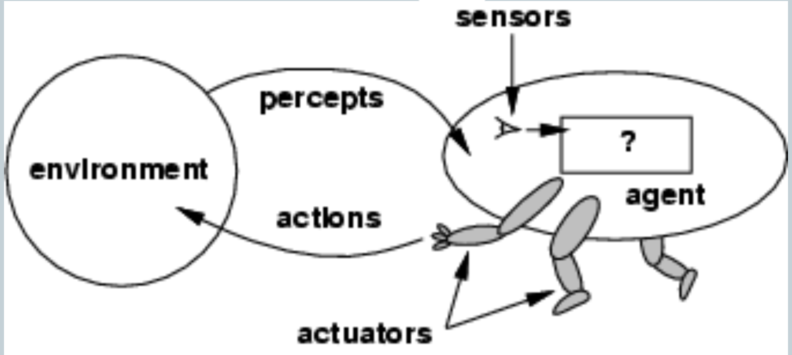
Agents

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

- Human agent:
  - eyes, ears, and other organs for sensors;
  - hands, legs, mouth, and other body parts for actuators

- Robotic agent:
  - cameras and infrared range finders for sensors
  - various motors for actuators

Agents and environments

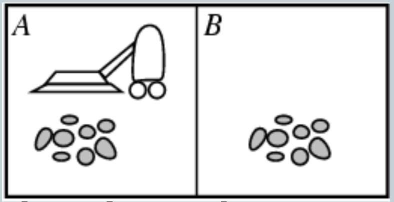


- The agent function maps from percept histories to actions:

$$[f: \mathcal{P}^\star \rightarrow \mathcal{A}]$$

- The agent program runs on the physical architecture to produce *f*
- agent = architecture + program

Vacuum-cleaner world



[Open Source Demo]

- Percepts: location and contents, e.g., [A,Dirty]
- Actions: *Left*, *Right*, *Suck*, *NoOp*
- *Agent's function → look-up table*
  - *For many agents this is a very large table*

Percept sequence	Action
$[A, Clean]$	$Right$
$[A, Dirty]$	$Suck$
$[B, Clean]$	$Left$
$[B, Dirty]$	$Suck$
$[A, Clean], [A, Clean]$	$Right$
$[A, Clean], [A, Dirty]$	$Suck$
⋮	⋮

Rational agents

- Rationality
  - Performance measuring success
  - Agents prior knowledge of environment
  - Actions that agent can perform
  - Agent’s percept sequence to date

- Rational Agent: For each possible percept sequence, a rational agent should select an action that is expected to *maximize its performance measure*, given
- the evidence provided by the percept sequence, and
- whatever built-in knowledge the agent has.

Rationality

- Rational is different from omniscience
  - Percepts may not supply all relevant information
  - E.g., in card game, don't know cards of others.

- Rational is different from being perfect
  - Rationality maximizes expected outcome while perfection maximizes actual outcome.

Autonomy in Agents

The **autonomy** of an agent is the extent to which its behaviour is determined by its own experience, rather than knowledge of designer.

- Extremes
  - No autonomy – ignores environment/data
  - Complete autonomy – must act randomly/no program
- Example: baby learning to crawl
- Ideal: design agents to have some autonomy
  - Possibly become more autonomous with experience

The PEAS Framework

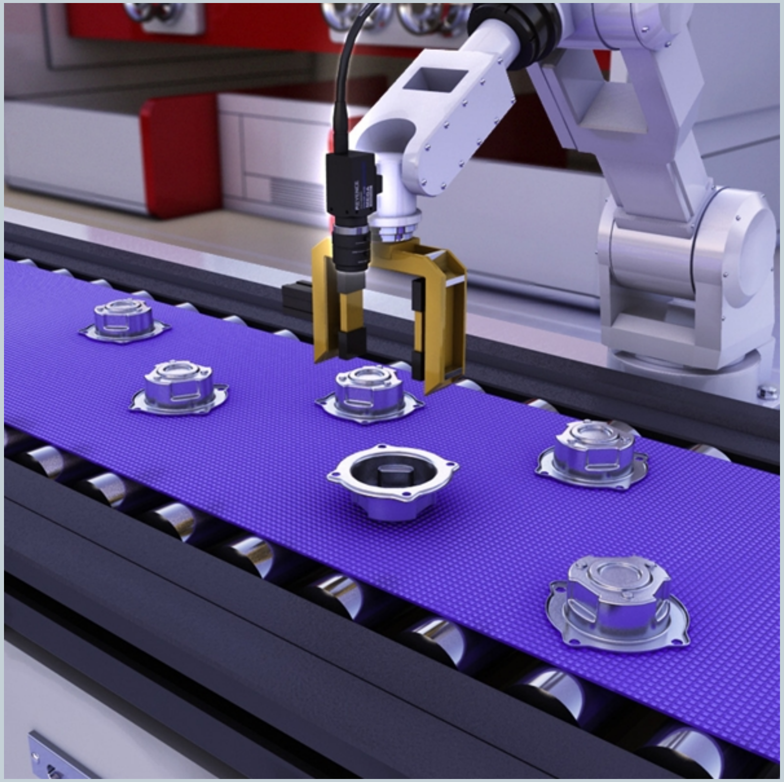
PERFORMANCE MEASURE, ENVIRONMENT, ACTUATORS, SENSORS

PEAS

- PEAS: Performance measure, Environment, Actuators, Sensors
- Specifies the setting for designing an intelligent agent

PEAS: Part-Picking Robot

- Agent: Part-picking robot
- Performance measure: Percentage of parts in correct bins
- Environment: Conveyor belt with parts, bins
- Actuators: Jointed arm and hand
- Sensors: Camera, joint angle sensors



PEAS

- Agent: Interactive Spanish tutor
- Performance measure: Maximize student's score on test
- Environment: Set of students
- Actuators: Screen display (exercises, suggestions, corrections)
- Sensors: Keyboard

Discussion: Self-Driving Car

- Performance measure:
- Environment:
- Actuators:
- Sensors:

Environments

Environment types

- Fully observable (vs. partially observable)
- Deterministic (vs. stochastic)
- Episodic (vs. sequential)
- Static (vs. dynamic)
- Discrete (vs. continuous)
- Single agent (vs. multiagent).

Fully observable (vs. partially observable)

- Is everything an agent requires to choose its actions available to it via its sensors?
  - If so, the environment is fully observable
- If not, parts of the environment are unobservable.
  - Agent must make informed guesses about world.

Cross Word	Poker	Backgammon	Taxi driver	Part picking robot	Image analysis
Fully	Partially	Fully	Partially	Partially	Fully

Deterministic (vs. stochastic)

- Does the change in world state depend *only* on current state and agent's action?
- Non-deterministic environments
  - Have aspects beyond the control of the agent
  - Utility functions have to guess at changes in world

Cross Word	Poker	Backgammon	Taxi driver	Part picking robot	Image analysis
Deterministic	Stochastic	Stochastic	Stochastic	Stochastic	Deterministic

Episodic (vs. sequential):

- Is the choice of current action
  - Dependent on previous actions?
  - If not, then the environment is episodic
- In sequential environments:
  - Agent has to plan ahead:
    - Current choice will affect future actions

Cross Word	Poker	Backgammon	Taxi driver	Part picking robot	Image analysis
Sequential	Sequential	Sequential	Sequential	Episodic	Episodic

Static (vs. dynamic):

- Static environments don’t change
  - While the agent is deliberating over what to do
- Dynamic environments do change
  - So agent should/could consult the world when choosing actions
- Semidynamic: If the environment itself does not change with the passage of time but the agent's performance score does.

Cross Word	**Poker**	**Backgammon**	**Taxi driver**	**Part picking robot**	**Image analysis**
Static	Static	Static	Dynamic	Dynamic	Semi

Another example: off-line route planning vs. on-board navigation system

Discrete (vs. continuous)

- A limited number of distinct, clearly defined percepts and actions vs. a range of values (continuous)

Cross Word	**Poker**	**Backgammon**	**Taxi driver**	**Part picking robot**	**Image analysis**
Discrete	Discrete	Discrete	Conti	Conti	Conti

Single agent (vs. multiagent):

- An agent operating by itself in an environment vs. there are many agents working together

Cross Word	Poker	Backgammon	Taxi driver	Part picking robot	Image analysis
Single	Multi	Multi	Multi	Single	Single

Discussion: Self-Driving Car

	Observable	Deterministic	Episodic	Static	Discrete	Agents
Self-Driving Car	partially	non-deterministic	sequential	dynamic	continuous	multi-agent

[Apple self-driving car was rear-ended by Nissan Leaf]

Summary.

	Observable	Deterministic	Episodic	Static	Discrete	Agents
Cross Word	Fully	Deterministic	Sequential	Static	Discrete	Single
Poker	Partially	Stochastic	Sequential	Static	Discrete	Multi
Backgammon	Fully	Stochastic	Sequential	Static	Discrete	Multi
Taxi driver	Partially	Stochastic	Sequential	Dynamic	Conti	Multi
Part picking robot	Partially	Stochastic	Episodic	Dynamic	Conti	Single
Image analysis	Fully	Deterministic	Episodic	Semi	Conti	Single

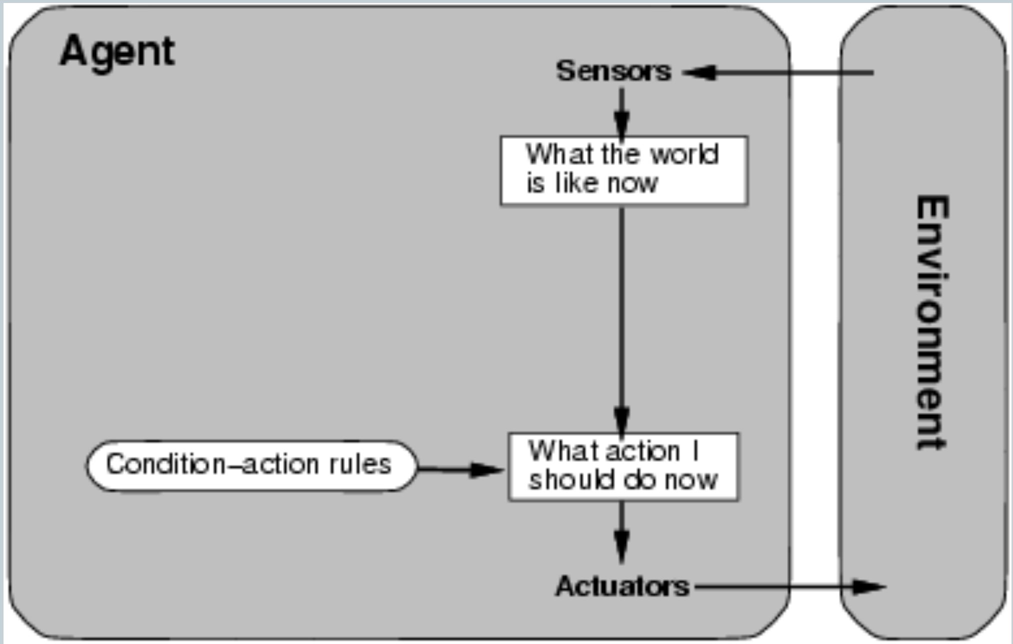
Agents

AGENT TYPES
LEARNING

Agent types

- Four basic types in order of increasing generality:
  - Simple reflex agents
  - Reflex agents with state/model
  - Goal-based agents
  - Utility-based agents
  - All these can be turned into learning agents

Simple reflex agents



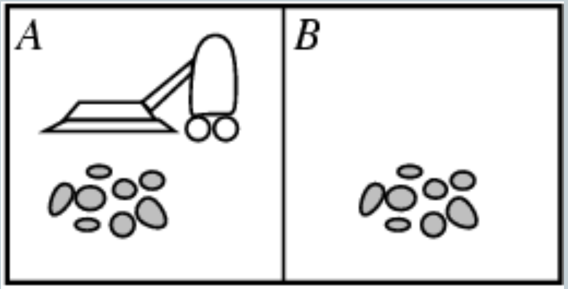
function REFLEX-VACUUM-AGENT([*location*,*status*]) **returns** an action

  if *status* = *Dirty* **then return** *Suck*
  else if *location* = *A* **then return** *Right*
  else if *location* = *B* **then return** *Left*

Vacuum Cleaner Reflex Agent

Robot forgets past, knows only current square

History	State	Action
[A,Clean]	[A,Clean]	Right
[A,Clean,Right ; B, Dirty]	[B,Dirty]	Suck
[A,Clean,Right ; B, Dirty, Suck; B, Clean]	[B, Clean]	Left
[A,Clean,Right ; B, Dirty, Suck; B, Clean, Left; A, Clean]	[A, Clean]	



Simple reflex agents

- Simple but very limited intelligence.
- **Action does not depend on percept history, only on current percept.**
    - Thermostat.
- Therefore no memory requirements.
- Infinite loops
    - Suppose vacuum cleaner does not observe location. What do you do given location = clean? Left on A or right on B -> infinite loop.
    - Fly buzzing around window or light.
    - Possible Solution: Randomize action.

States: Beyond Reflexes

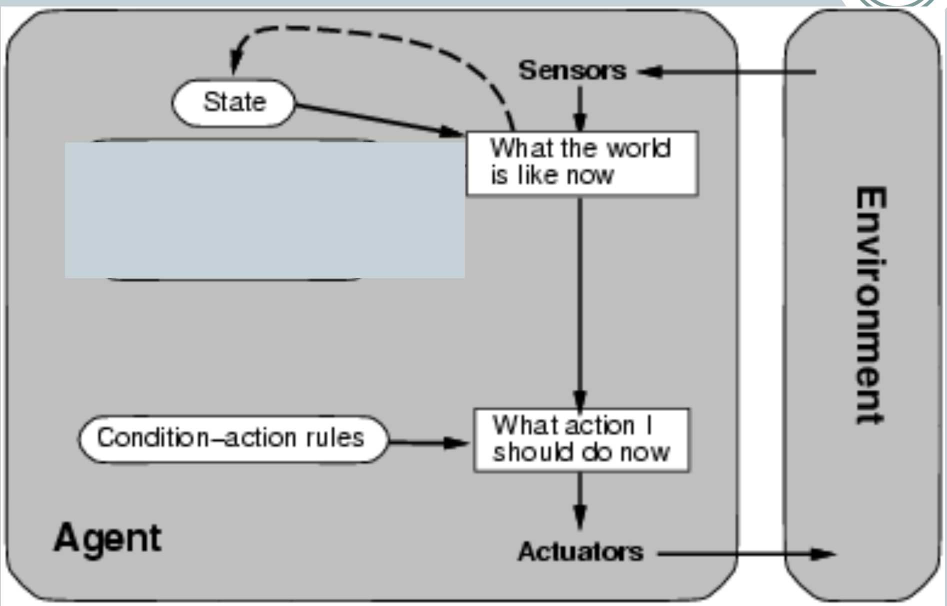
- Recall the agent function that maps from percept histories to actions:

[f: $\mathcal{P}^* \rightarrow \mathcal{A}$]

- An agent program can implement an agent function by maintaining an **internal state** (memory)
  - e.g. cell phone knows its battery usage
- The internal state can contain information about the state of the external environment.
- The state depends on the history of percepts and on the history of actions taken:

[f: $\mathcal{P}^*, \mathcal{A}^* \rightarrow \mathcal{S} \rightarrow \mathcal{A}$] where $\mathcal{S}$ is the set of states.

State-based reflex agents

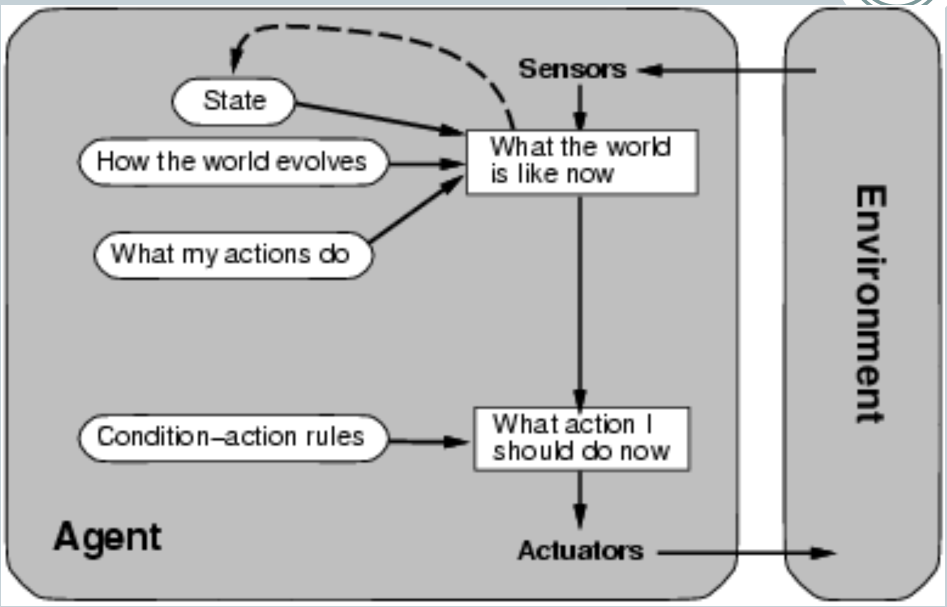


- Update state = remember history
- Many (most?) state-of-the-art systems for open-world problems follow this architecture
  - (e.g. translation)
- No thinking

```
function REFLEX-AGENT-WITH-STATE(percept) returns action
   static: state, a description of the current world state
           rules, a set of condition-action rules

   state ← UPDATE-STATE(state, percept)
   rule ← RULE-MATCH(state, rules)
   action ← RULE-ACTION[rule]
   state ← UPDATE-STATE(state, action)
   return action
```

Model-based reflex agents



- Know how world evolves
  - Overtaking car gets closer from behind
- Predict how agents actions affect the world
  - Wheel turned clockwise takes you right
- Model-based agents predict consequences of their actions

state ← UPDATE-STATE(state,action,percept,model)

~~*state* ← UPDATE-STATE(*state*, *percept*)~~

rule ← RULE-MATCH(*state*, *rules*)

action ← RULE-ACTION[*rule*]

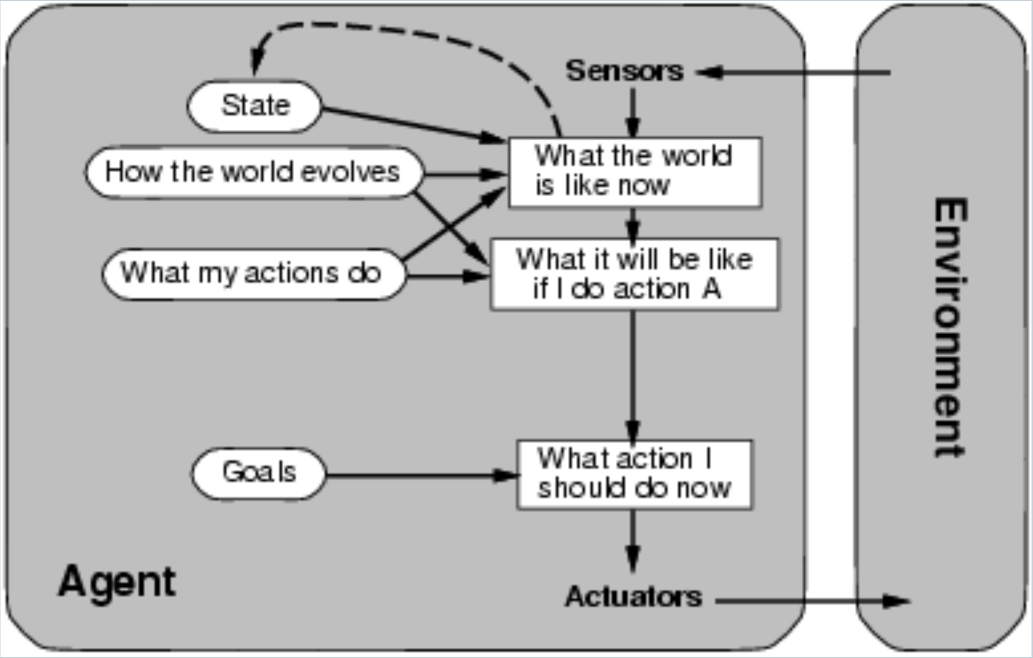
state ← UPDATE-STATE(*state*, *action*)

return *action*

Goal-based agents

- knowing state and environment? Enough?
  - Car can go left, right, straight
- Has a goal
  - A destination to get to
- Uses knowledge about a goal to guide its actions
  - E.g., Search, planning

Goal-based agents



- Reflex agent brakes when it sees brake lights. Goal based agent reasons
  - Brake light -> car in front is stopping -> I should stop -> I should use brake

Example

- The Monkey and Banana Problem
- Monkeys can use a stick to grasp a hanging banana

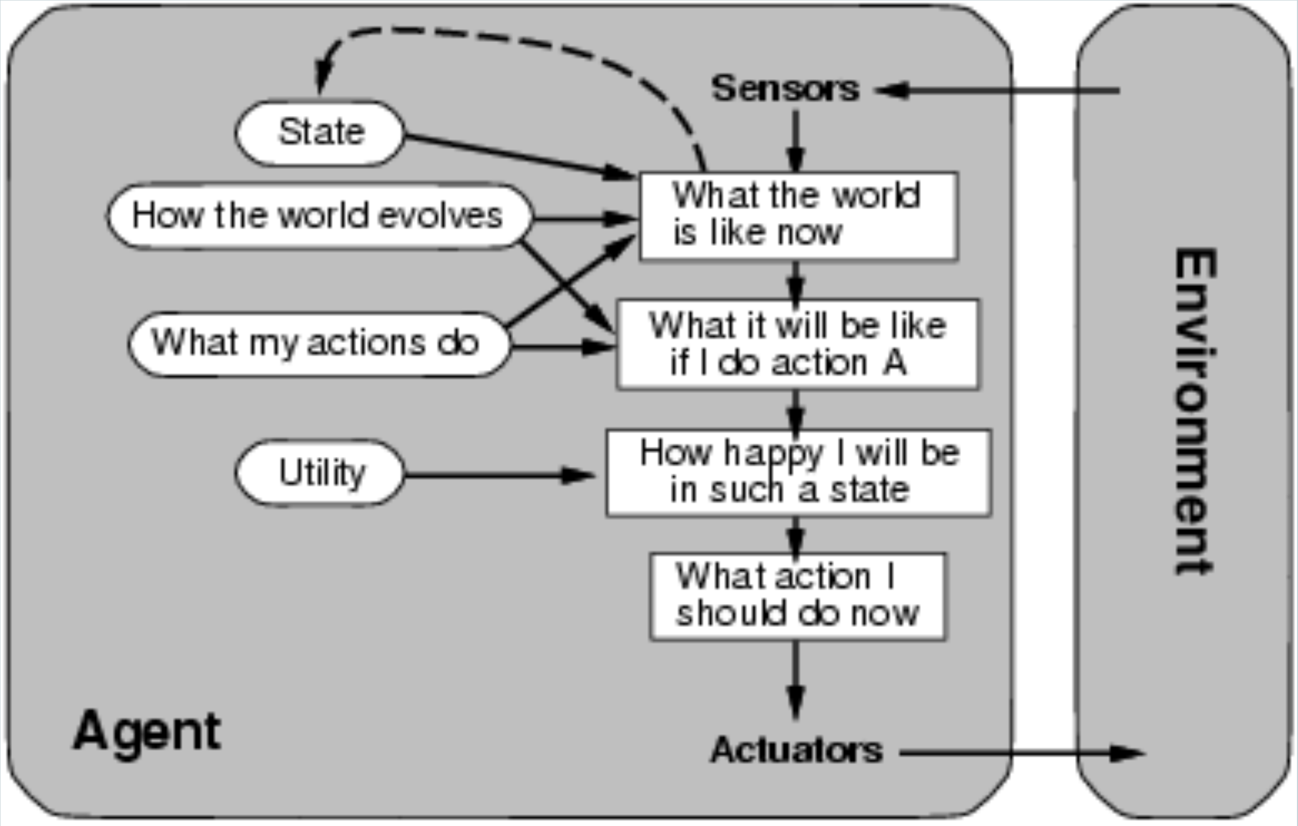
Utility-based agents

- Goals are not always enough
  - Many action sequences get car to destination
  - Consider other things. How fast, how safe.....
- A utility function maps a state onto a real number which describes the associated degree of "happiness", "goodness", "success".
- Where does the utility measure come from?
  - Economics: money.
  - Biology: number of offspring.
  - Your life?

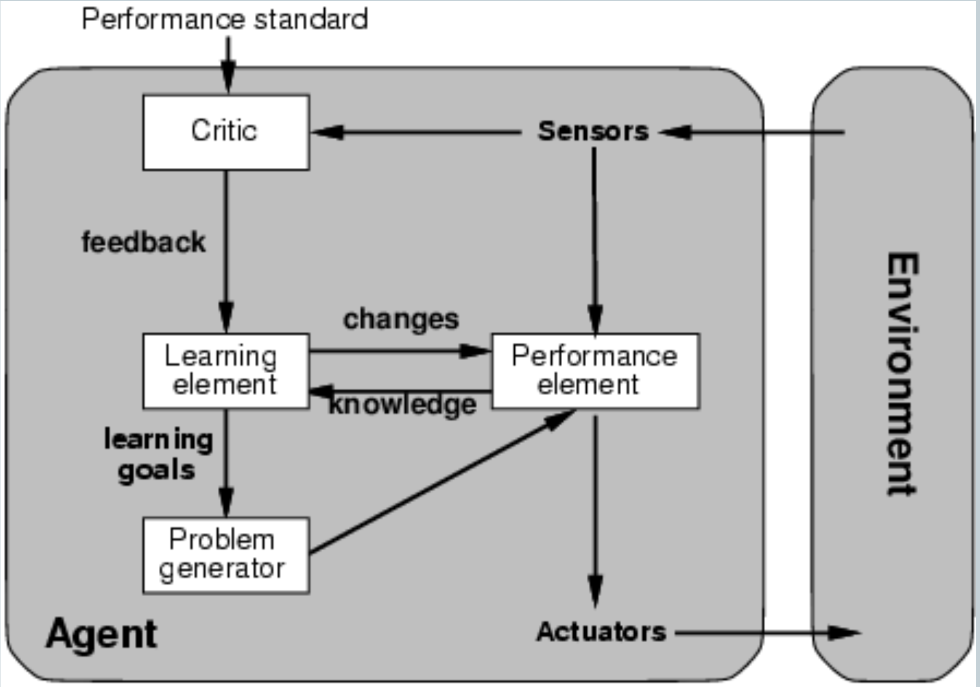
Utility for Self-Driving Cars

- What is the performance metric?
- Safety - No accidents
- Time to destination
- What if accident is unavoidable? E.g.
  - is it better to crash into an old person than into a child?
  - How about 2 old people vs. 1 child?

Utility-based agents



Learning agents

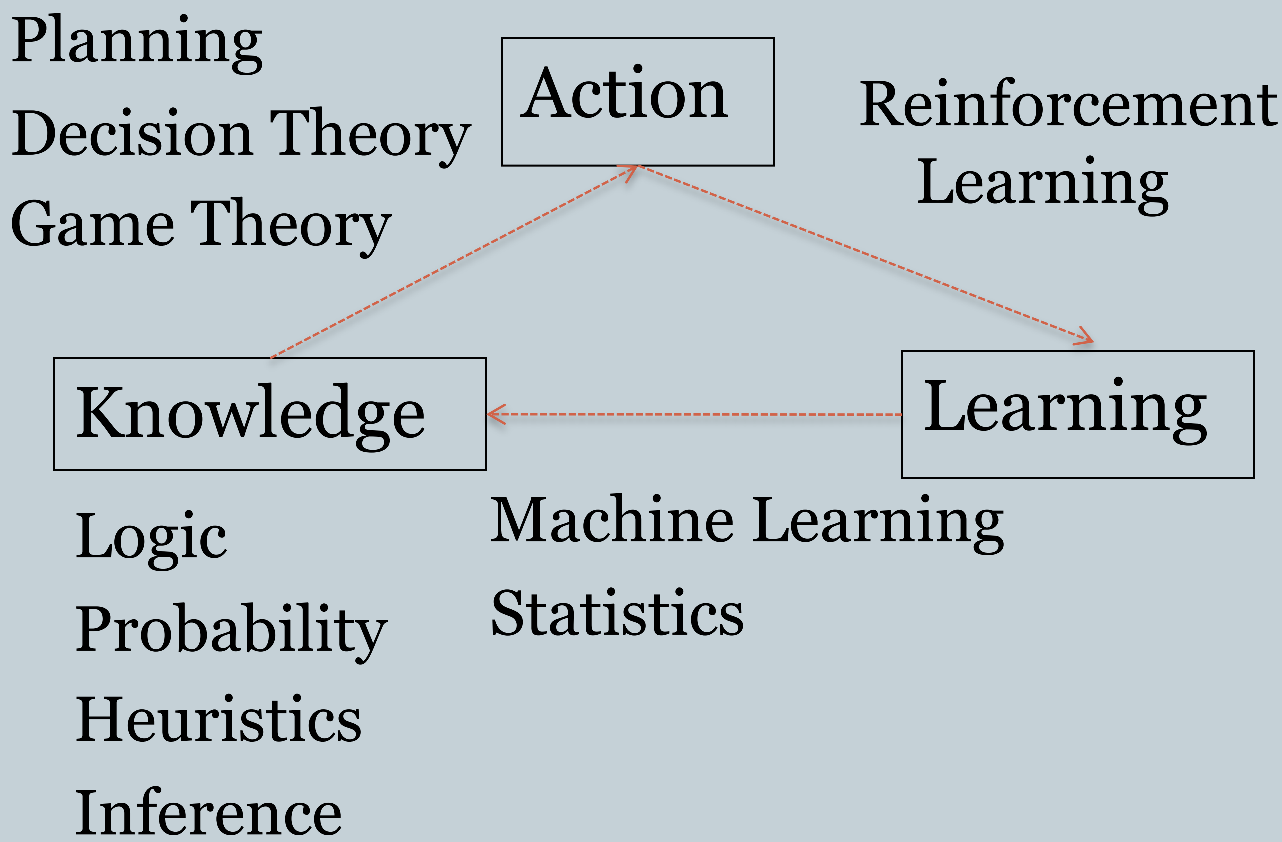


- Performance element is what was previously the whole agent
  - Input sensor
  - Output action
- Learning element
  - Modifies performance element.

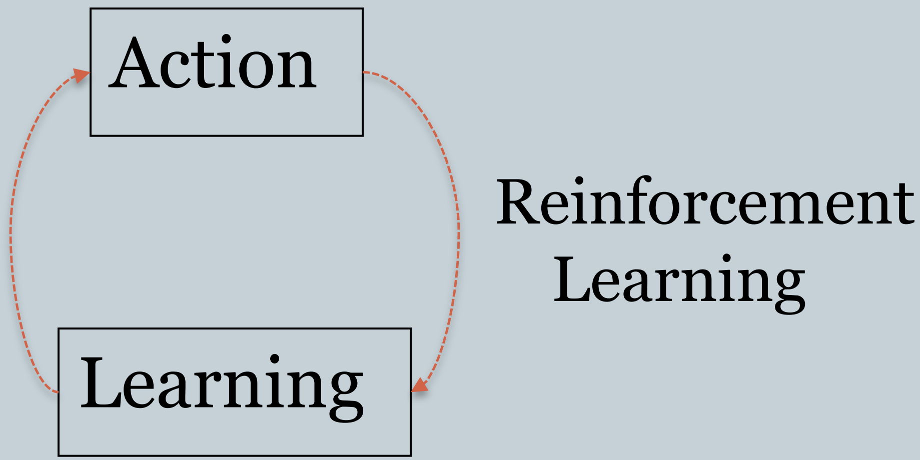
Learning agents (Self-Driving Car)

- Performance element
  - How it currently drives
- Actuator (steering): Makes quick lane change
- Sensors observe
  - Honking
  - Sudden Proximity to other cars in the same lane
- Learning element tries to modify performance elements for future
  - Problem generator suggests experiment: try out something called Signal Light
- Exploration vs. Exploitation
  - Exploration: try something new
  - \+ Improved Performance in the long run
  - – Cost in the short run

The Big Picture: AI for Model-Based Agents



The Picture for Reflex-Based Agents



- Studied in AI, Cybernetics, Control Theory, Biology, Psychology.
- Skinner box

Discussion Question

- Model-based reasoning has a large overhead.
- Our large brains are very expensive from an evolutionary point of view.
- Why would it be worthwhile to base behaviour on a model rather than "hard-code" it?
- For what types of organisms in what type of environments?
  - The dodo is an example of an inflexible animal

Summary

- Agents can be described by their PEAS.
- Environments can be described by several key properties: 64 Environment Types.
- A rational agent maximizes the performance measure for their PEAS.
- The performance measure depends on the **agent function**.
- The **agent program implements** the agent function.
- 4 main **architectures** for agent programs.
- In this course we will look at some of the common and useful combinations of environment/agent architecture.