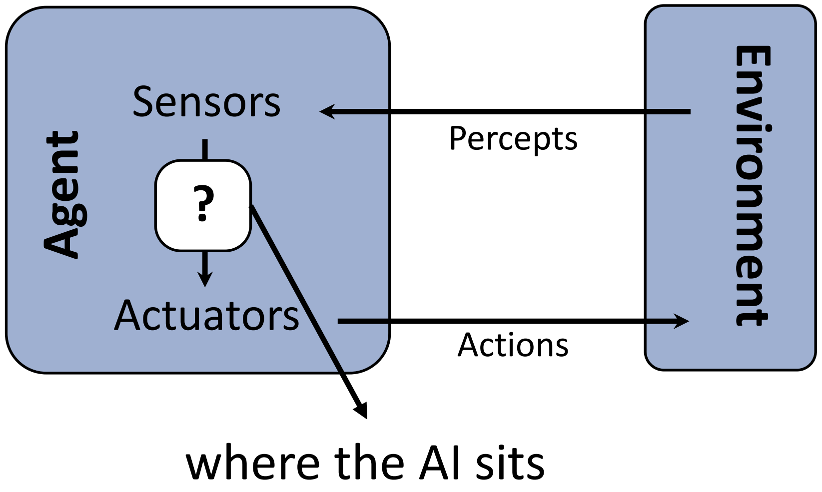
COMP 341: Introduction to AI
Agents

Asst. Prof. Barış Akgün

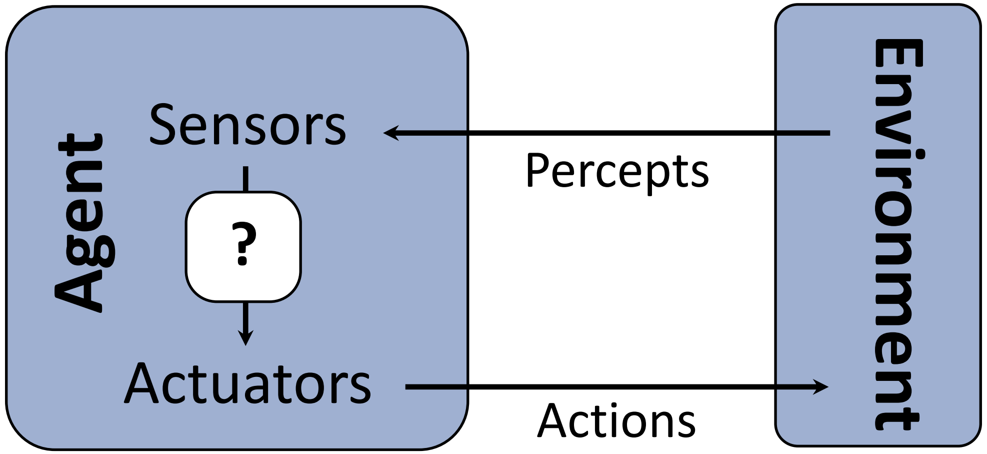
Koç University

Recap

- AI: Science of making *agents* that **act rationally**
- Agent: An autonomous entity that exists in some kind of *environment* and that *perceives* and *acts*
- Rationality: Maximizing **expected utility** ([interesting read])
- Rational Agent: An agent which selects actions to maximize its expected utility



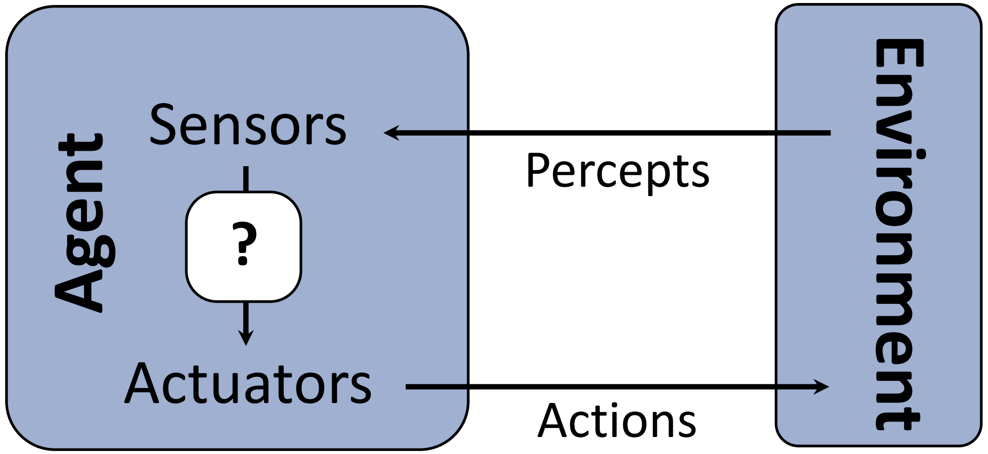
Agents



- An agent maps percept histories to actions – but based on what?
- **AI**: The science of making **agents** that act **rationally**
- Rational Agent: An agent that acts to maximize its utility
- Is perfect rationality achievable?
    - Philosophical question, is maximizing expected utility rational? (read)

How do we formulate an AI problem?

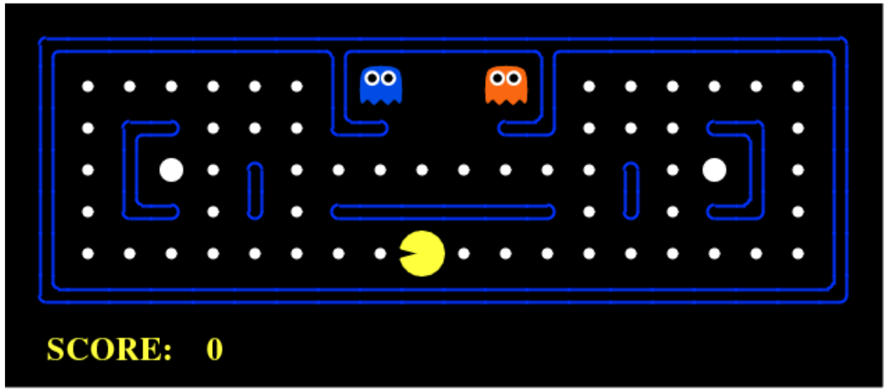
- Performance/Utility
  - Where does this come from?
- Environment
- Actuators/Actions
- Sensors/Percepts

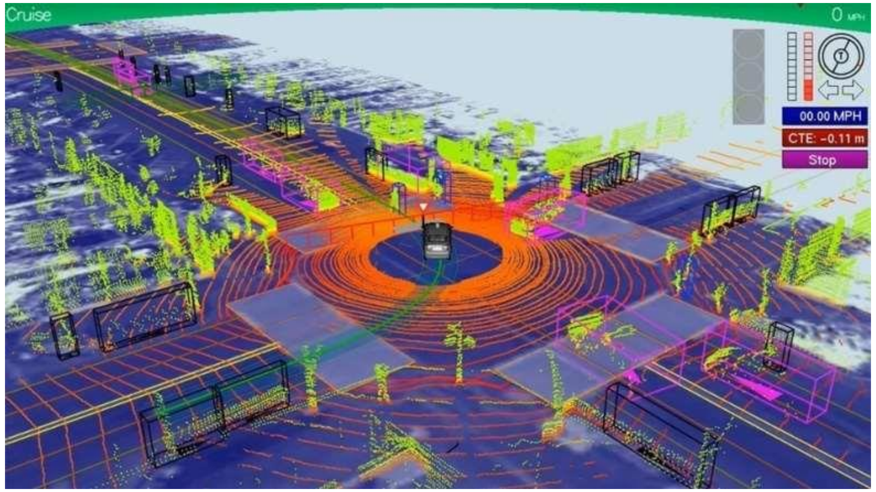


"The Environment"

- The agent is in some environment and is a part of it
- How to represent the environment? Specifically for the purposes of the agent or for the given problem

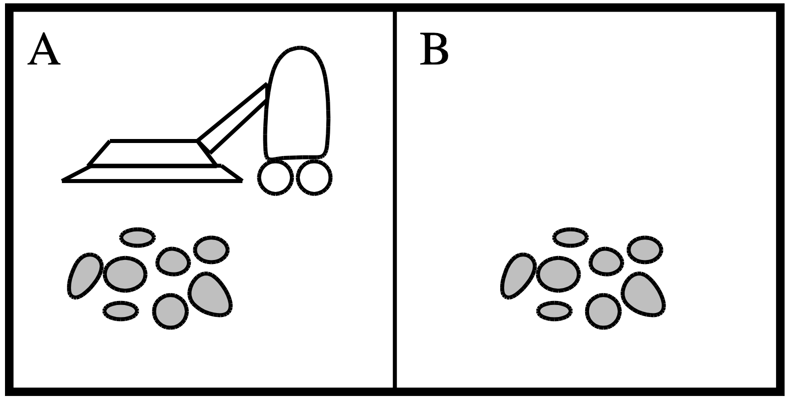
- For example, finding a path from A to B in İstanbul
  - Do the roads matter?
  - Does the temperature, humidity etc. matter?
  - Do the buildings matter?





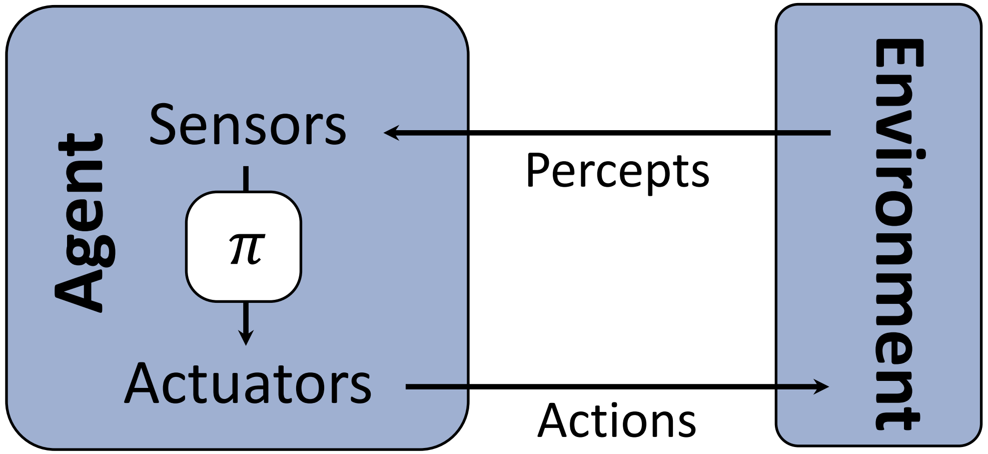
The State

- **Abstraction**: Remove/do not collect any unneeded details
- State is an **abstracted** description of the environment



- States?
- Actions?

Why is AI Hard?

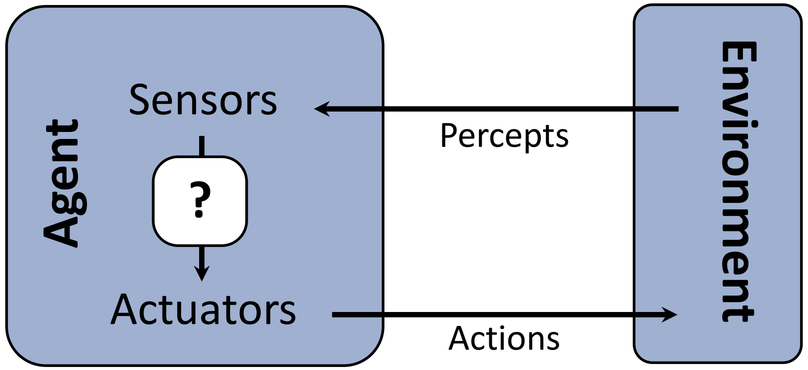


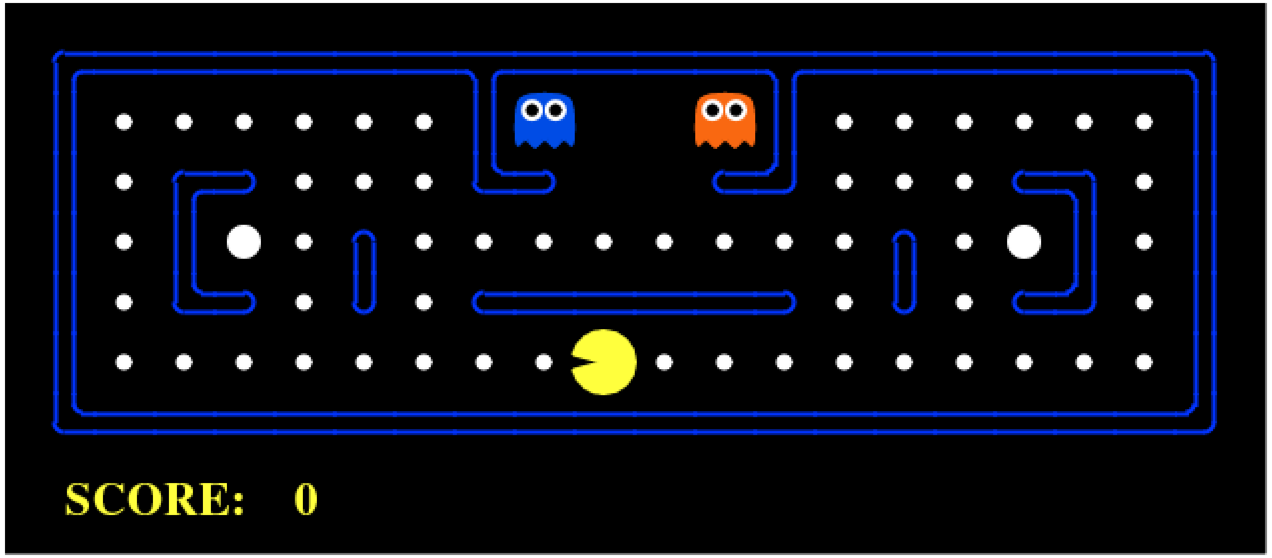
$$\pi: H \to A$$
$$H = (S_0, S_1, \dots, S_t)$$
$$\max(E[U])$$

- (S): State in the world that the agent can perceive/build through its sensory inputs
  - (H): State/Percept history
- (A): A set of actions that the agent can do through its actuators
- AI: Find π to maximize the utility! – How?
- (S x A), let alone (H x A), can be prohibitively large!
- This class: General methods/techniques to represent π for a variety of problems

This assumes that the problem is already formulated, which is also not always easy!

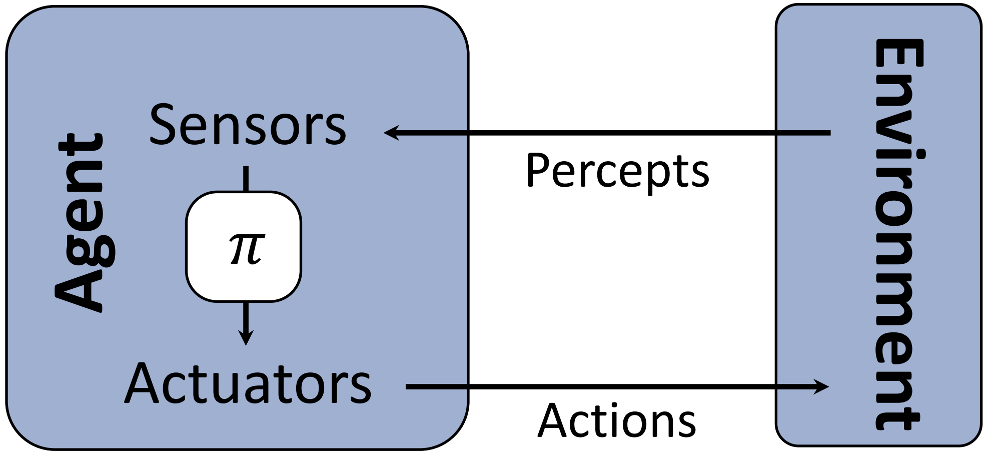
Example: Pac-Man as an Agent





How can we create a Pacman agent?

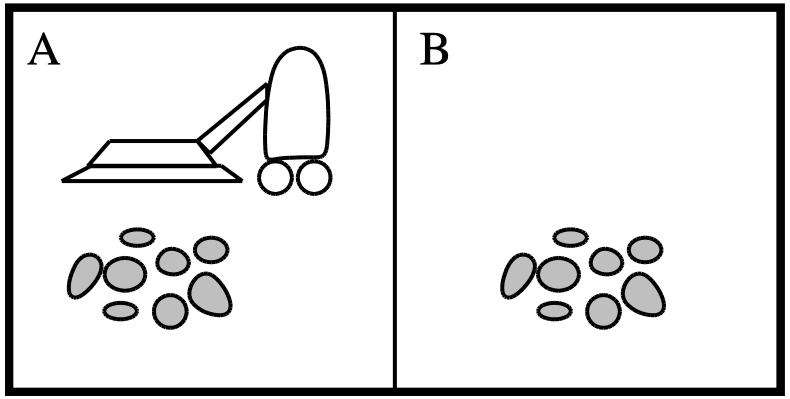
Agents



$$\pi: H \to A$$
$$H = (S_0, S_1, \dots, S_t)$$
$$\max(E[U])$$

How to program π?

- Look up table?
- Rules?
- Functions?
- Goals?

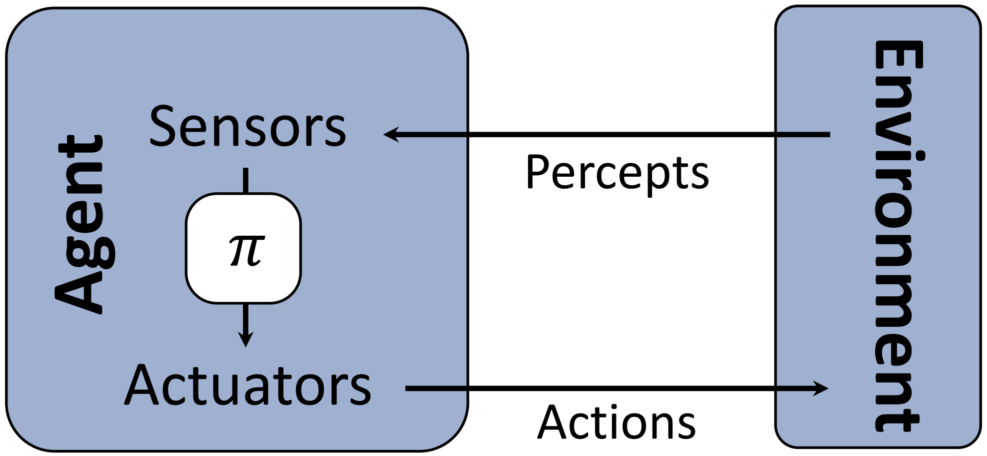


π?

Agent Types

- Reflex:
  - Simple reflex
  - Model-based

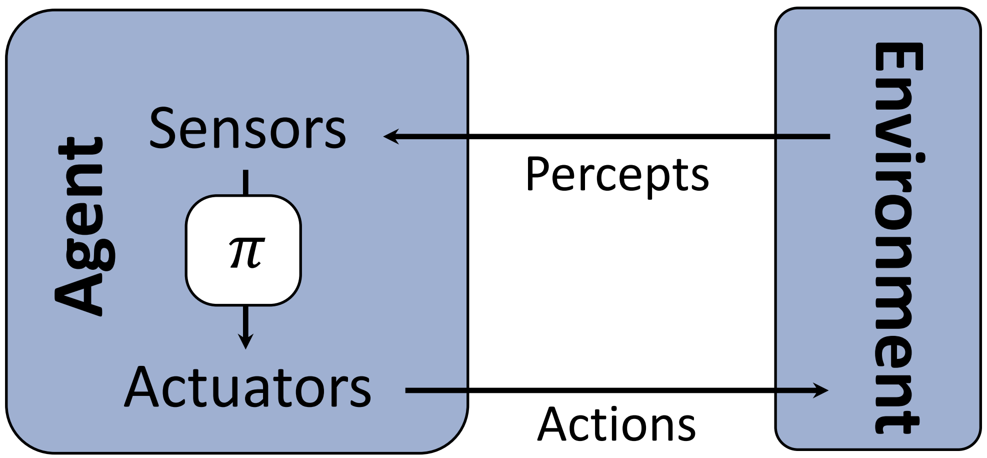
- Planning:
  - Goal-based
  - Utility-based



- how the environment is **VS** how the environment would be

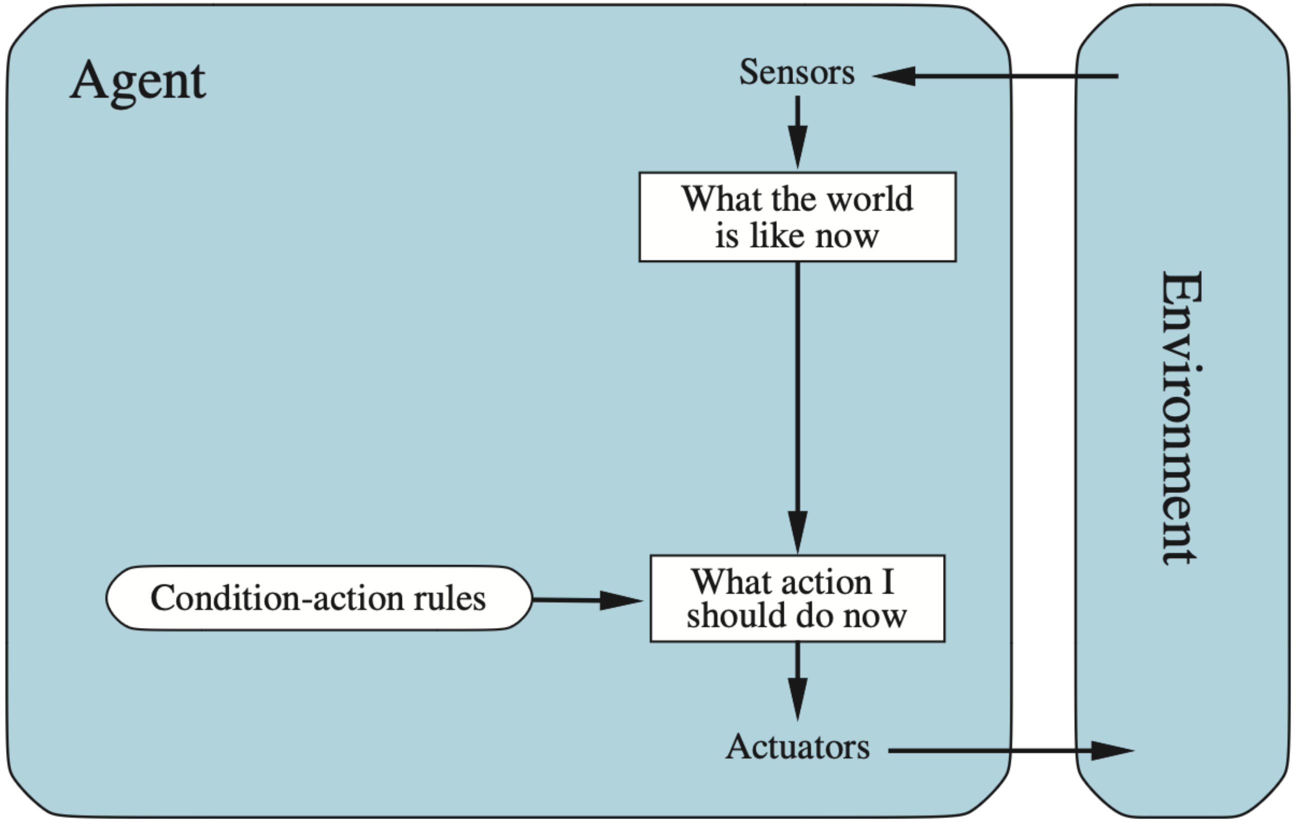
Agent Types

- Reflex:
  - Simple reflex
  - Model-based



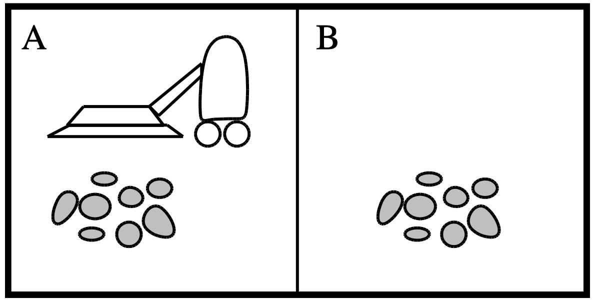
- how the environment is: current state to action
  - Look up table, rules
  - Control Systems ($u = ke + d\dot{e}$)
  - …

Simple Reflex Agent



Sensor information directly determines the state

A Simple Reflex Agent



State: <location, status>
Action Set: {Move, Suck}

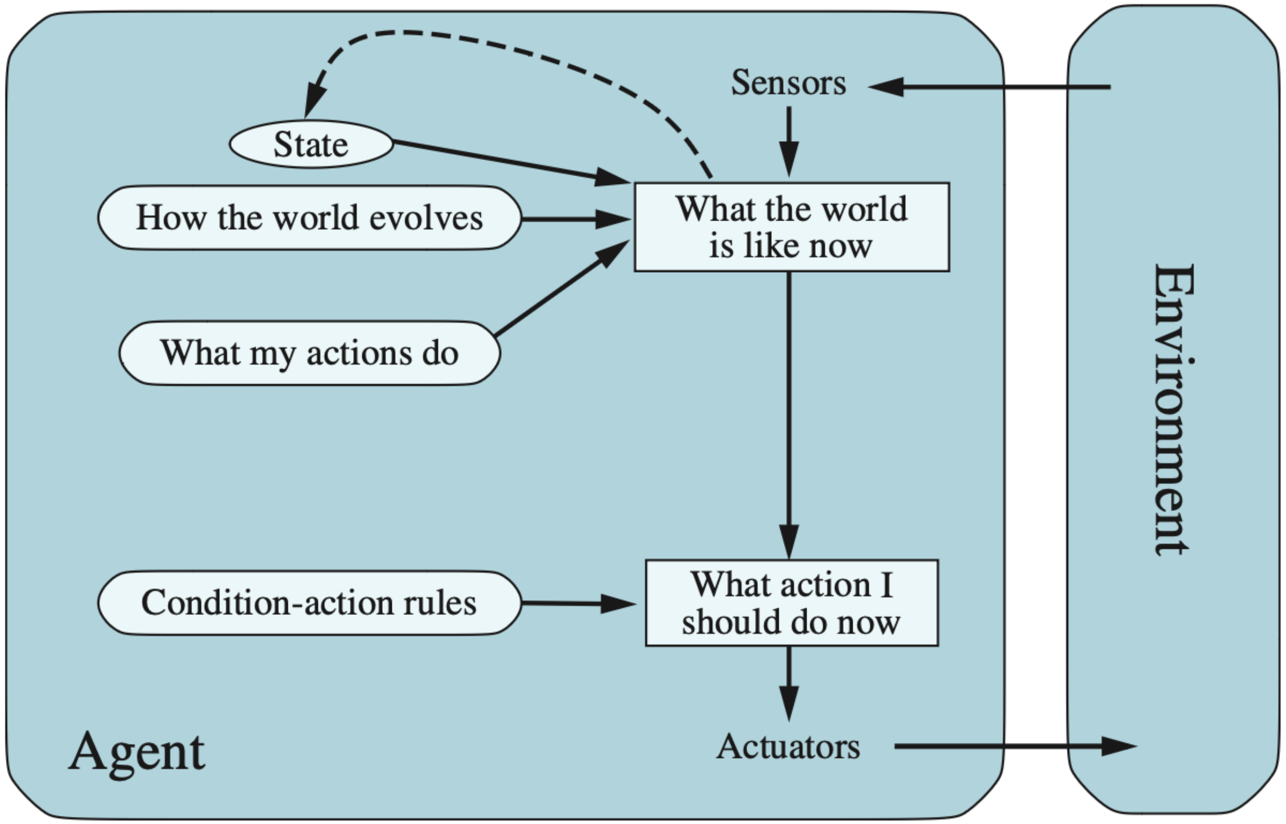
Any ideas on the behavior?

State	Action
A, clean	Move
A, dirty	Suck
B, clean	Move
B, dirty	Suck

VS

```
IF (dirty) THEN:
   Suck
ELSE:
   Move
```

Model-based Reflex Agent

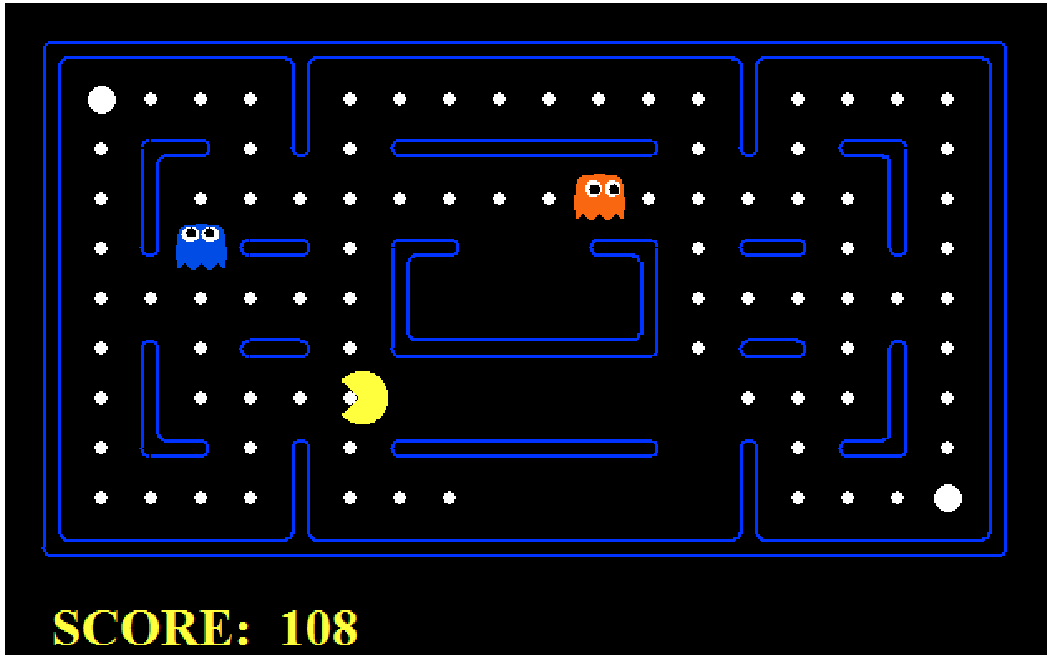


State: Sensors + **world model** (how the world evolves and consequences of agent's actions)

Especially useful when the agent cannot perceive everything!

Pacman: Model-Based Reflex Agent

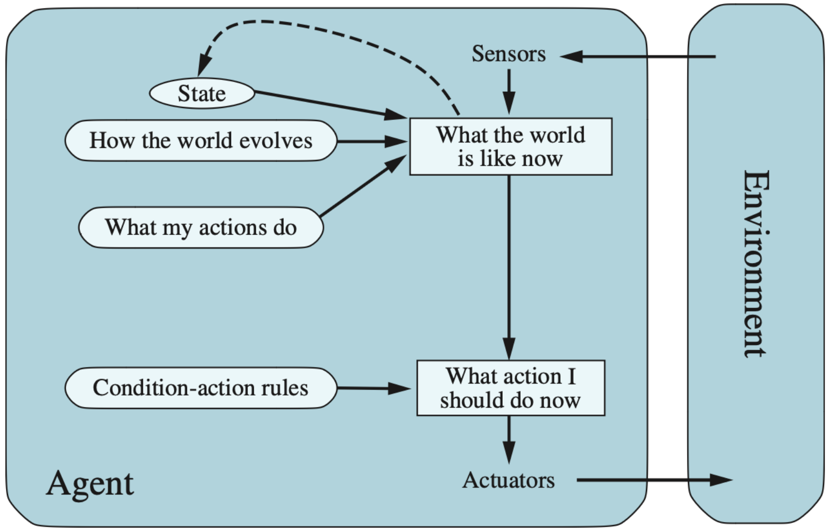
- Current state: s(t)
  - Maze Info (food, power capsules, empty space, walls)
  - Ghost Info (locations and scaredness duration)
  - Pacman Info (location)
  - Score
- Actions: a
  - North, South, East, West, Stop
- World Model: T
  - How will the state change if I apply a certain action? $s(t+1) = T(s(t),a)$
  - What about the ghosts? (assume they stay where they are, for now)

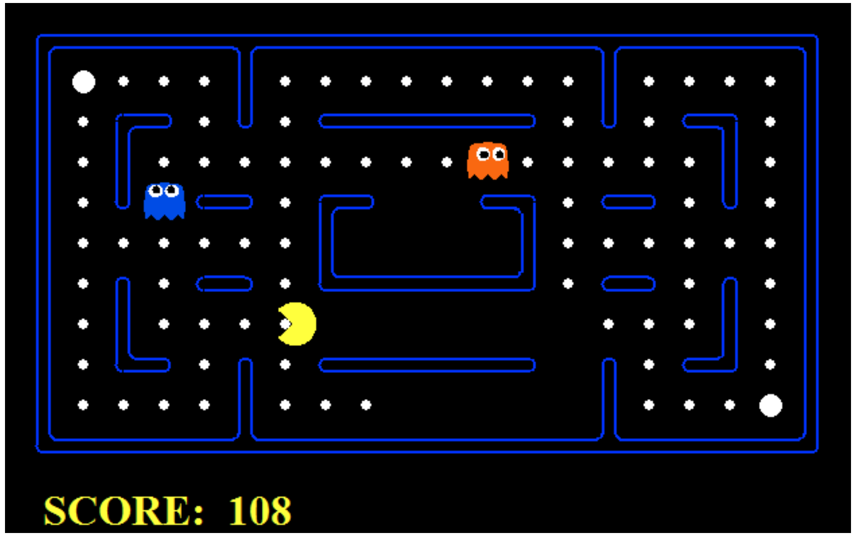


- How to select an action at a given state s?

Will be in one of your projects!

How to select the next action?





- "Imagine" the next states for all actions using the world model
- Evaluate the resulting states
- Chose the action that leads to the best next state!

Let’s design an evaluation function

- Evaluation function: Gets a state and returns a number, indicating the “goodness” of a state

- Let
  - x: state
  - $f_i(x)$: i’th feature extracted from state x

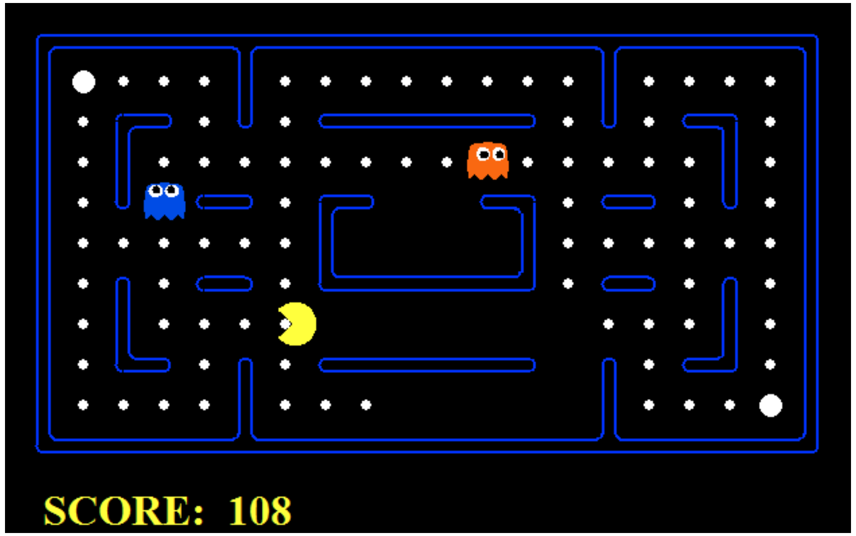
$$J(x) = w_1 f_1(x) + w_2 f_2(x) + \ \ldots + w_n f_n(x) = \sum_{i=1}^{n} w_i f_i(x)$$

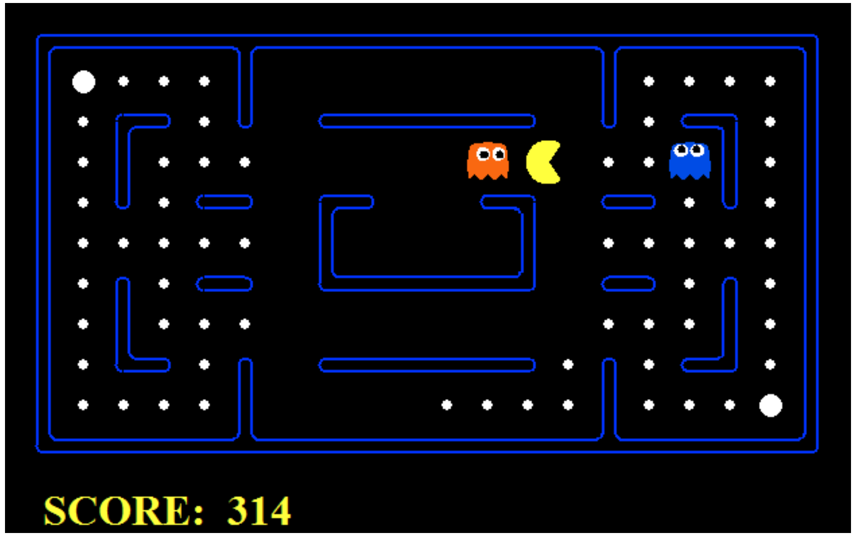
State Evaluation - Features

- Trivial: Delta Score
- Distance to food, capsules
- Distance to ghosts
- Whether and how long ghosts are edible
- Etc.

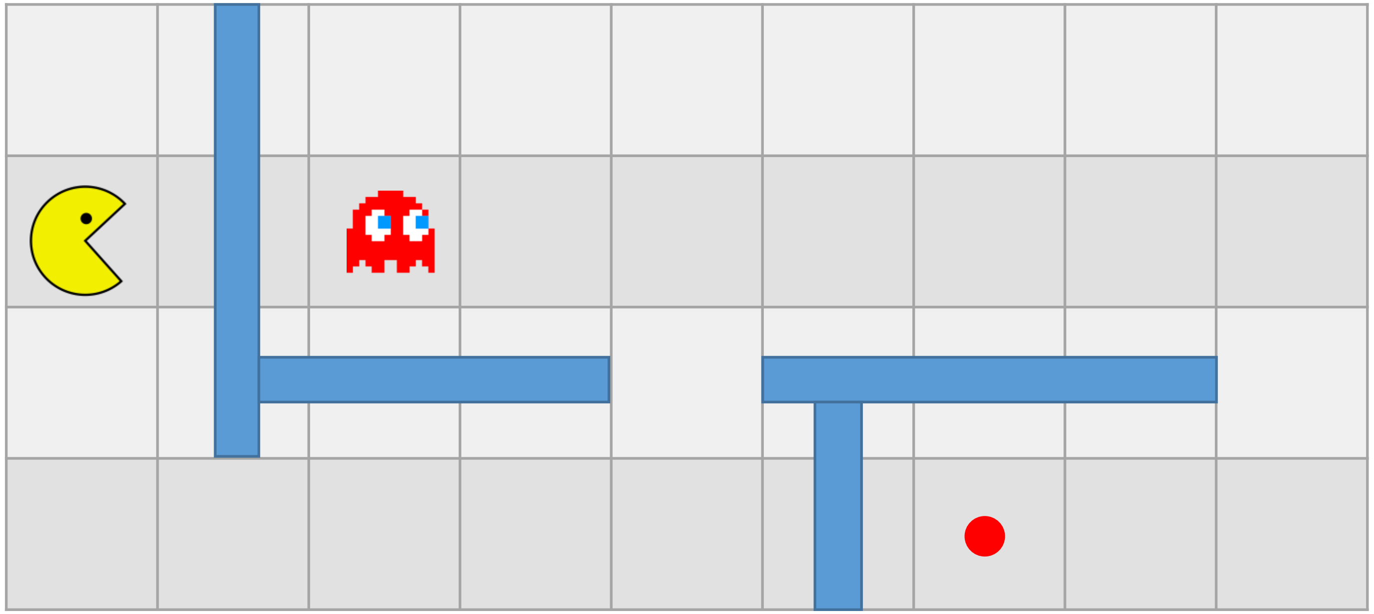
- Let’s call each of these **features**

Which one is preferable?





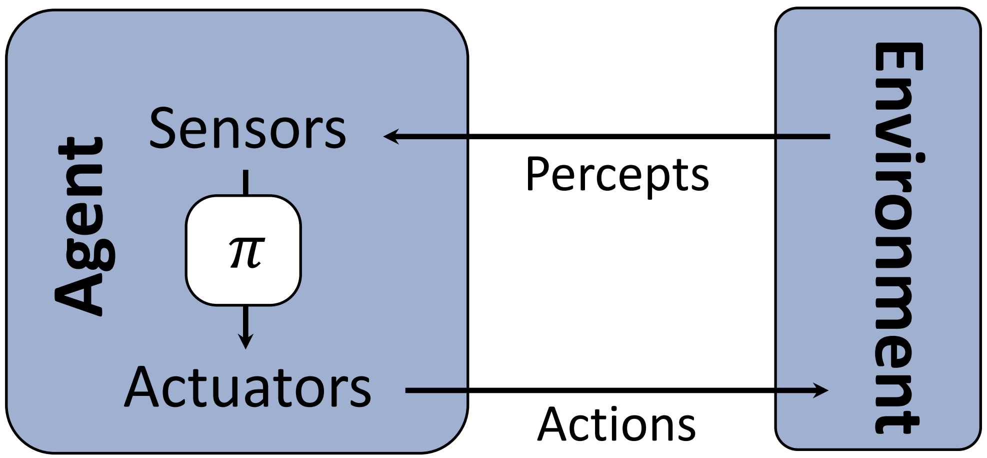
Distances



To the ghost?
To the red dot?

Agent Types

- Reflex:
  - Simple reflex
  - Model-based
- Planning:
  - Goal-based
  - Utility-based

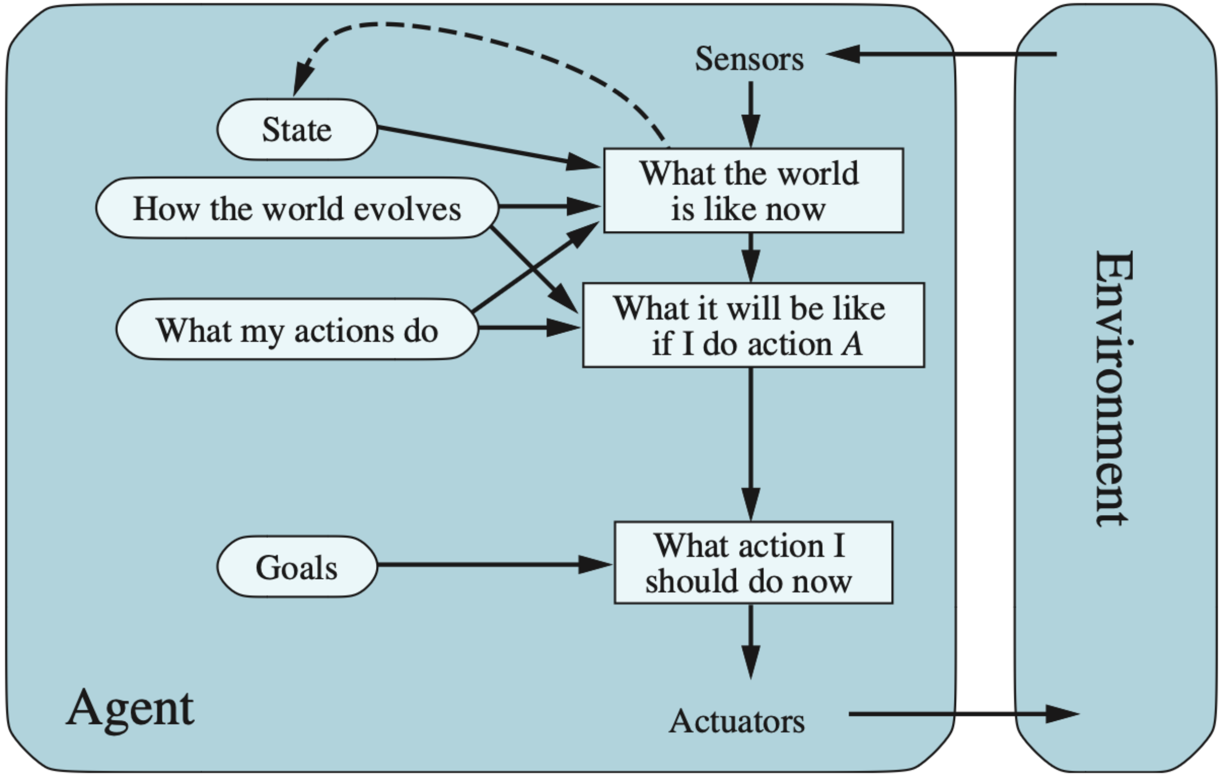


- how the environment is: current state to action

VS

- how the environment would be: current state + future states to action

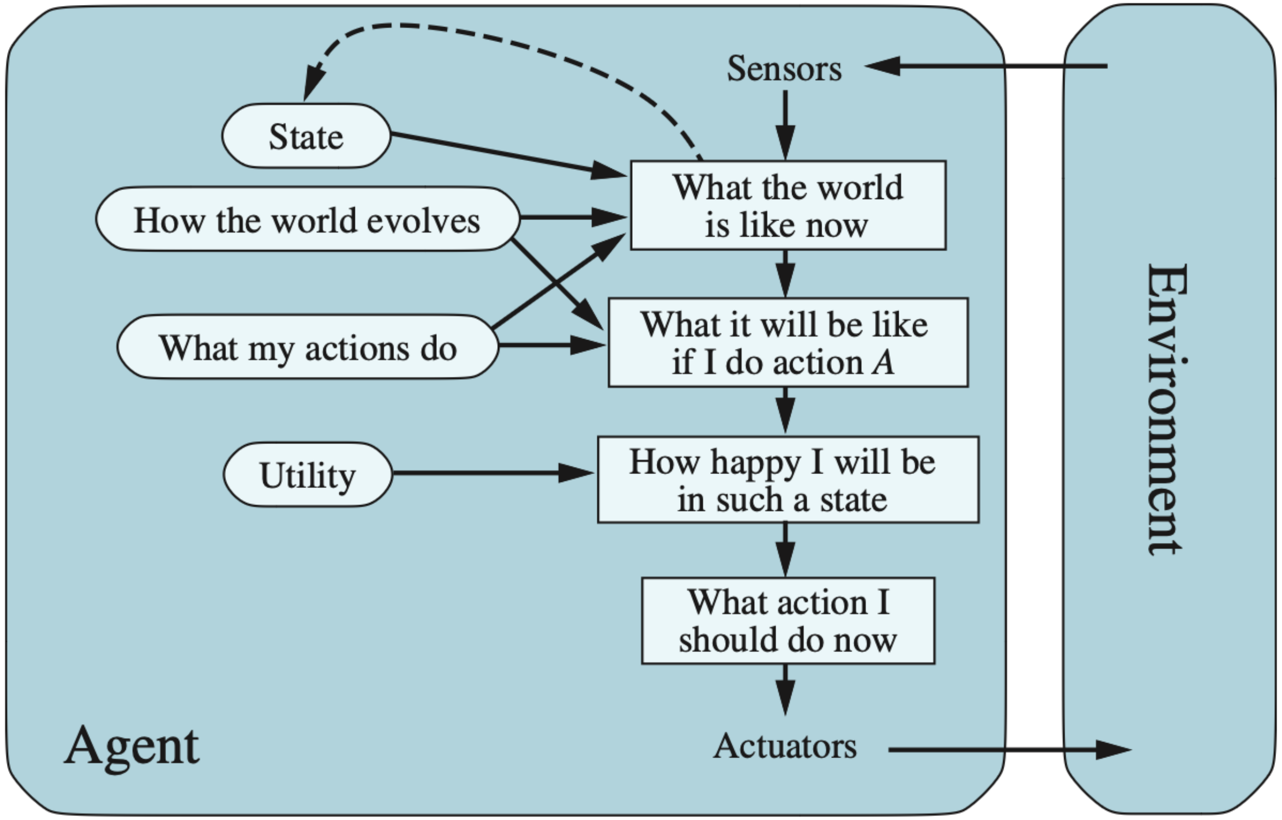
Goal Based Agent



Plans to see if the it is getting closer/accomplishing the goal. Only the goal matters.
Needs a model! (why?)

GPS Navigation

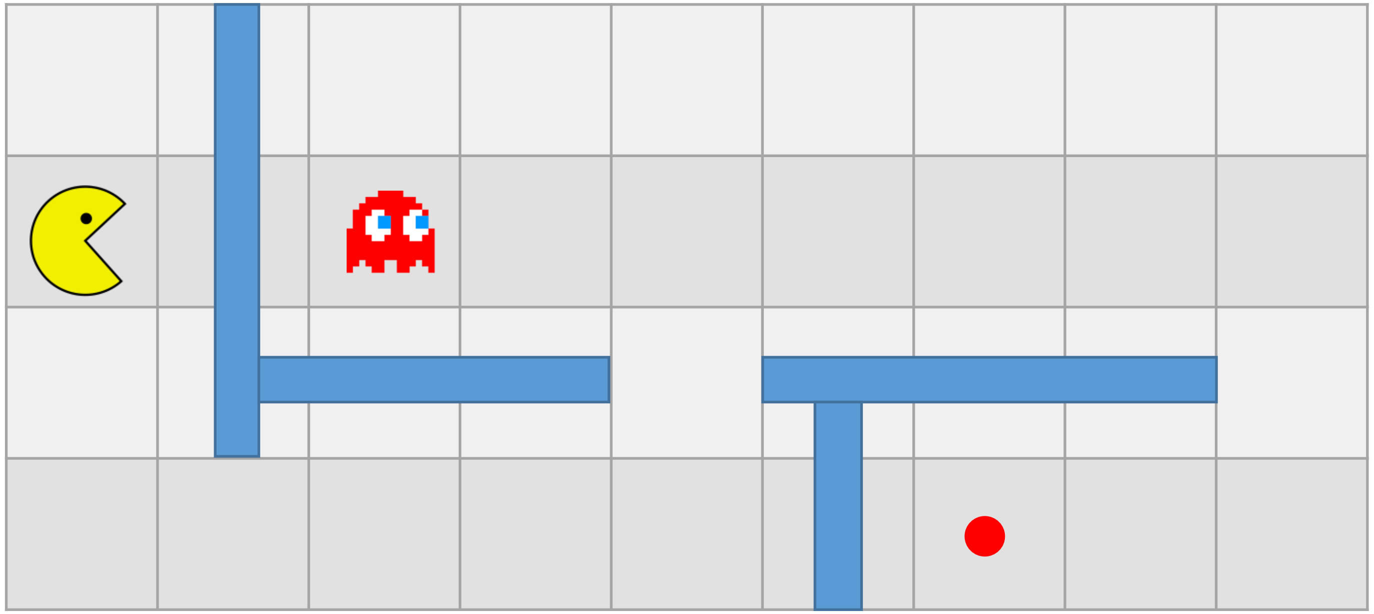
Utility Based Agent



In addition to planning, tries to get higher quality behavior as dictated by the **utility** function i.e. in addition to goals

Planning to go from point A to point B using the shortest path or the most pleasant path

Distances – How?



How to calculate the distances?

A Learning Agent Architecture

- Previous agents “exist” but how?
  - Engineered, Learning, Mixed …

(This is not the best figure in my opinion but the book authors aim to be as general as possible)