CS5100 Foundations of Artificial Intelligence

Module 01 Lesson 02

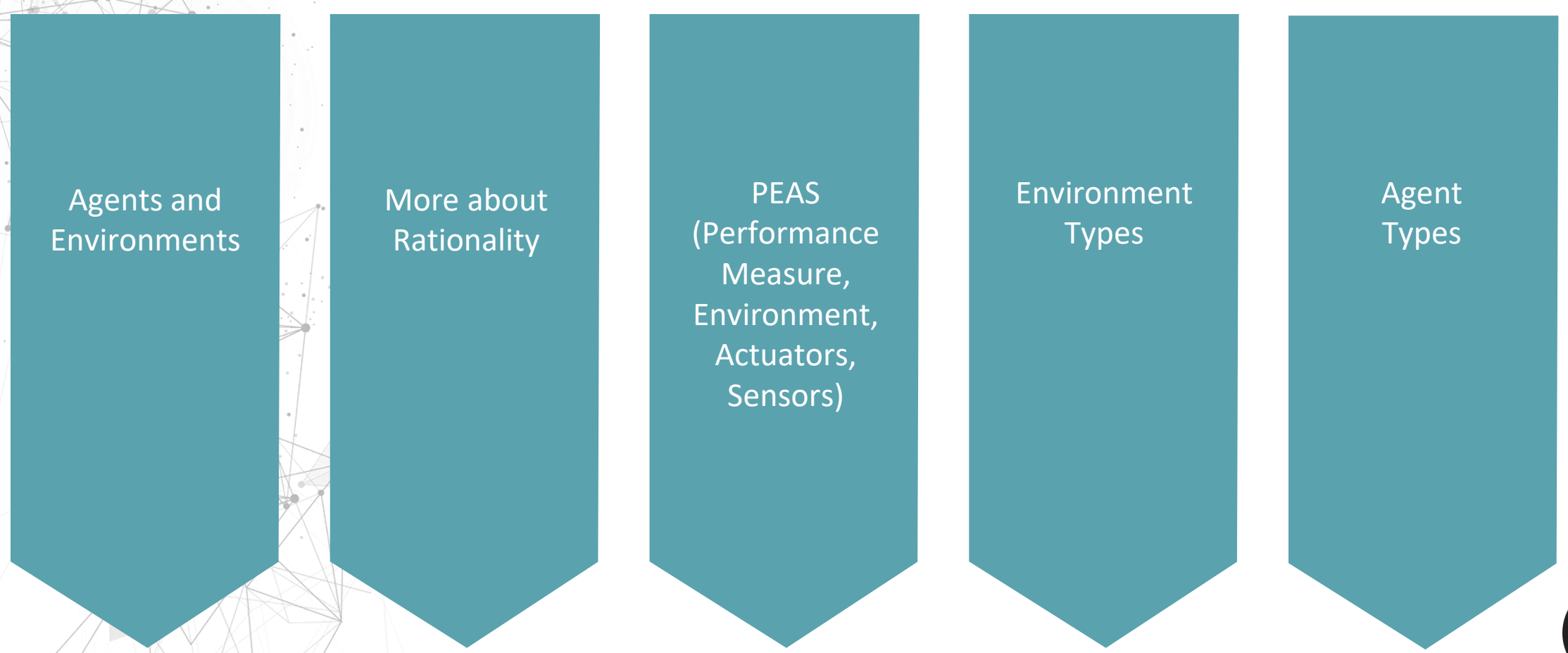
Intelligent Agents

Some images and slides are used from CS188 UC Berkeley/AIMA with permission
All materials available at http://ai.berkely.edu / http://aima.cs.berkeley.edu

Overview
A framework for the rest of the course

Why this framework:

Using rationality to develop design principles for successful ‘intelligent’ agents.



Overview

What are Agents?

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

The Concept of Rationality

We are using the notion of agents to analyze (AI) systems.

We're not trying to divide the world into agents and non-agents.

In the previous lecture, we said we're going to be looking at rational agents in this course.

In the next few slides, we'll further define this concept to understand AI systems better.

Task Environments & PEAS

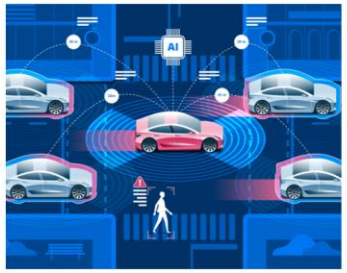
Talking about rational agents, we talk about

Performance measures, Environment,
Actuators, Sensors,

PEAS description !

PEAS description of automated taxi?

Task environments are like
problem descriptions,
where rational agents are solutions!



Task Environments Properties

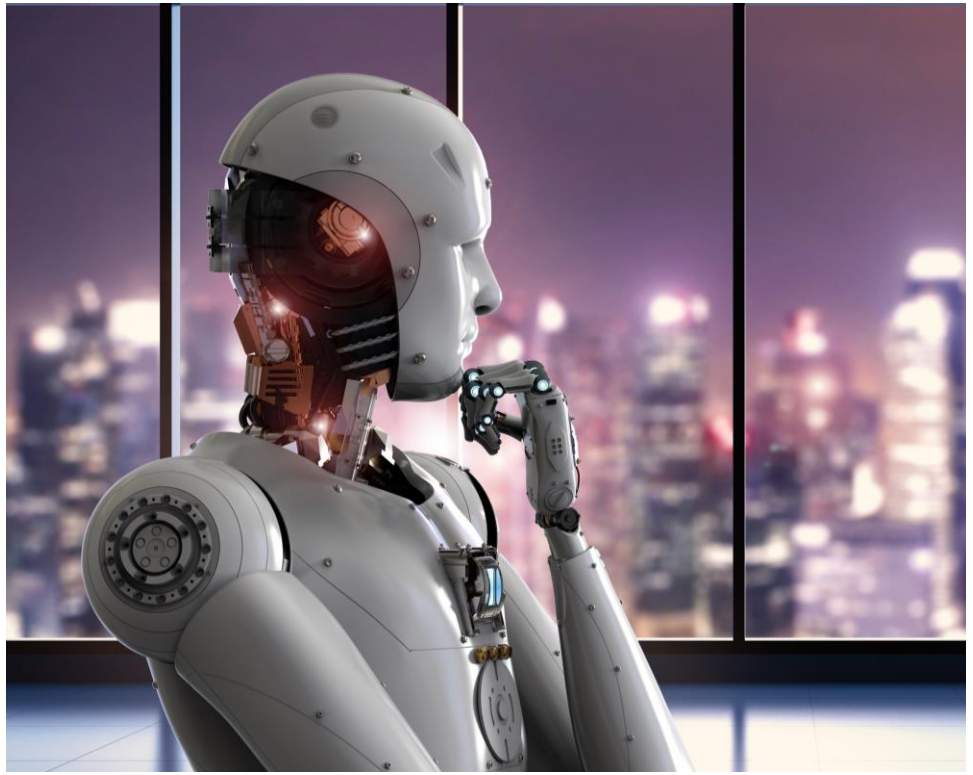
Structure of Agents

Agent = architecture + program

percepts from sensors → **agent program** → action choices → actuators

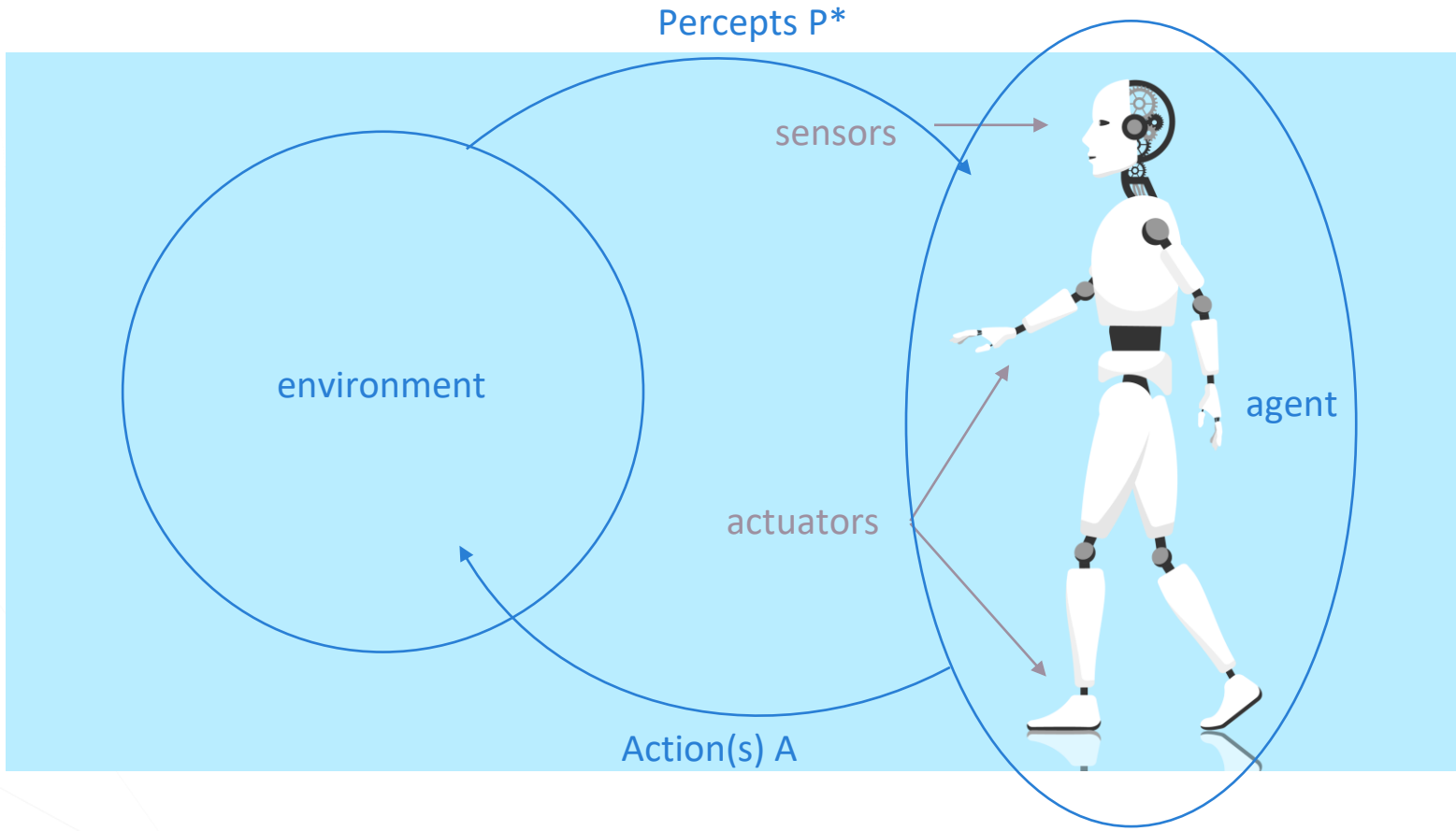
What are Agents?

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

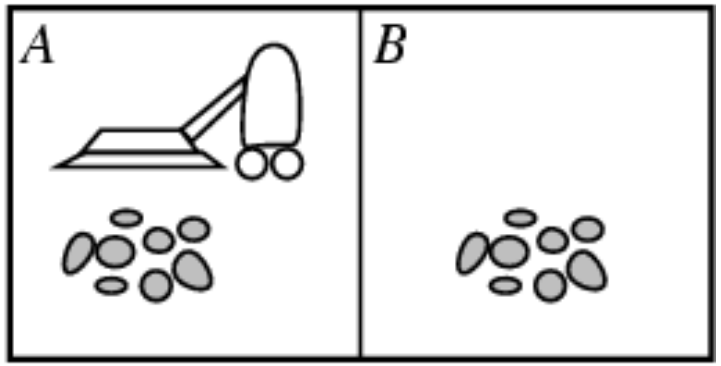


Agents and Environments

$[f: P^* \rightarrow A]$

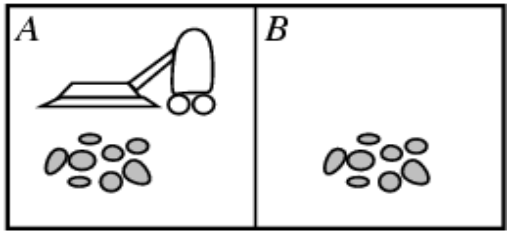


Vacuum-Cleaner World



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

A Vacuum-Cleaner Agent



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*

The Concept of Rationality

We are using the notion of agents to analyze (AI) systems.

We're not trying to divide the world into agents and non-agents.

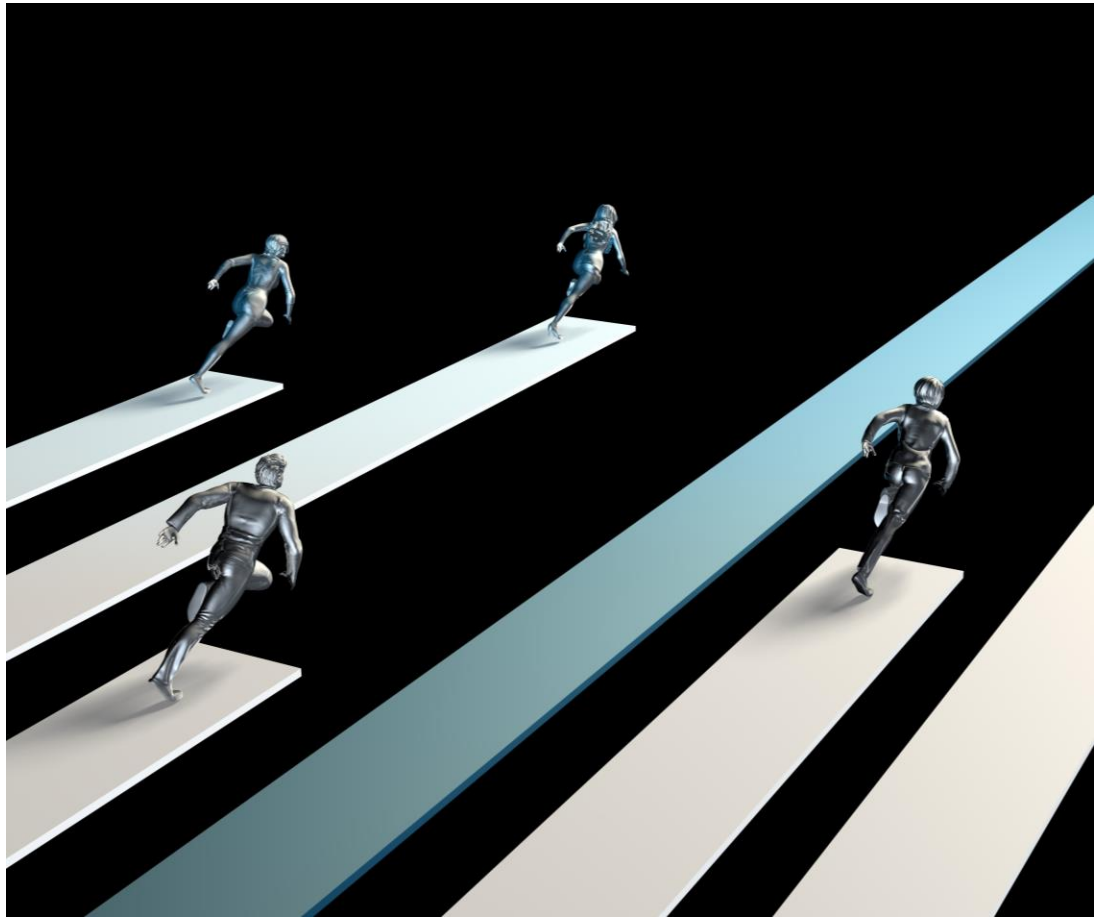
In the previous lecture, we said we're going to be looking at rational agents in this course.

In the next few slides, we'll further define this concept to understand AI systems better.

Rational Agents .. 1

- Rational agents try to do the right thing, but what *is* the right thing?
- How do we define agent success?
- Look at sequence of **environment** states, consequences of agent behavior
- Results of actions taken by the agent, reacting to its percepts
- To define success:
  - need objective performance measure
  - based on what we need in the world,
  - not how we think the agent should behave.
  - Will lead to tradeoffs

- What is success in the vacuum world?
- What is success in a course? ☺



Rationality

What's rational depends on:

- the performance measure that defines success,
- agent's percept sequence thus far,
- agent's prior knowledge of the environment,
- actions the agent can perform

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

Rational Agents .. 2

Percepts may not supply all relevant info

Agents have to explore environment

Vacuum cleaner has to check if clean

Agents should learn and adapt from experience

→ makes them autonomous

Rationality ➔ agent has the capacity to learn, explore, adapt, be autonomous

But first let's define task environments

Task Environments & PEAS

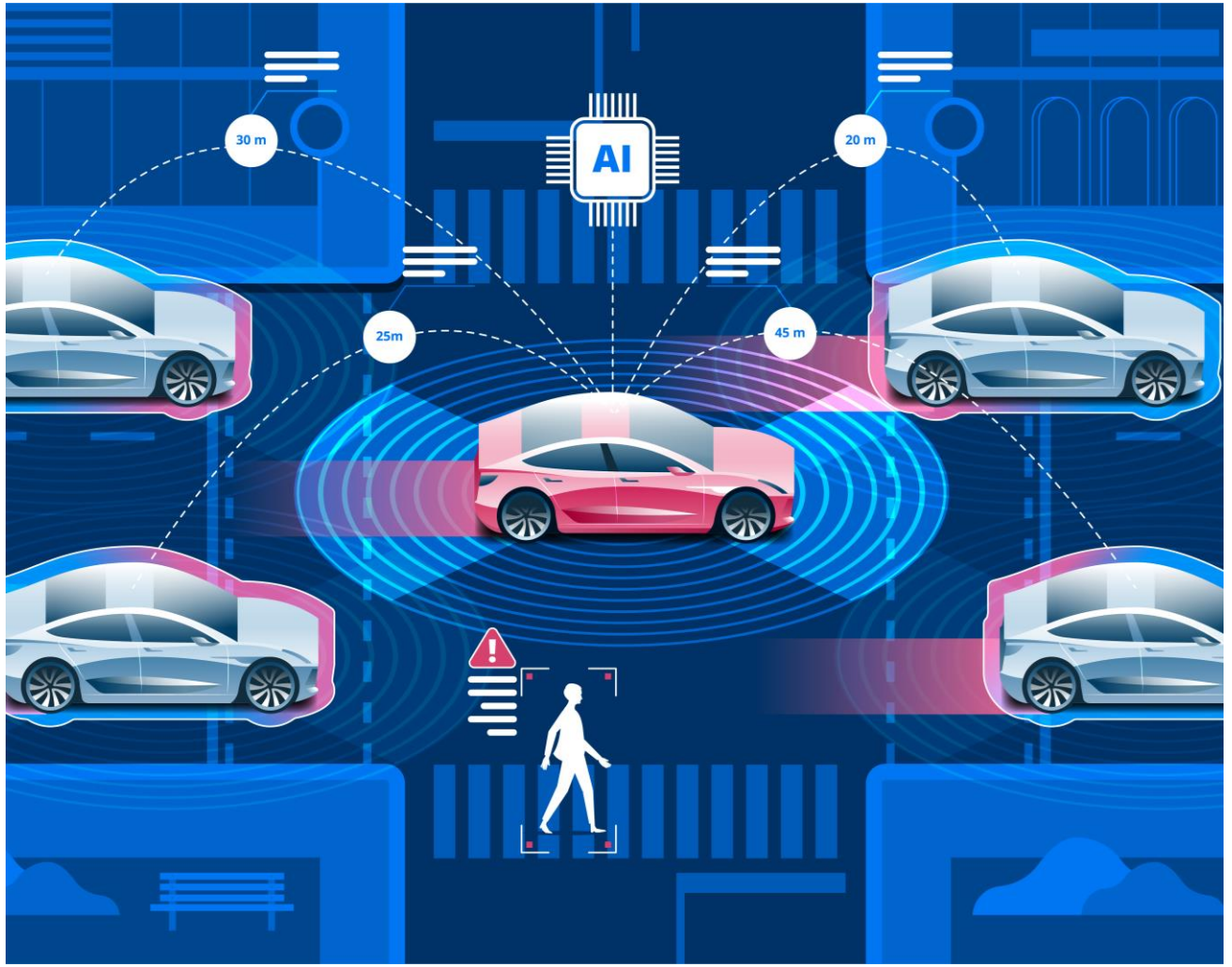
Talking about rational agents, we talk about

Performance measures, Environment, Actuators, Sensors,

PEAS description !

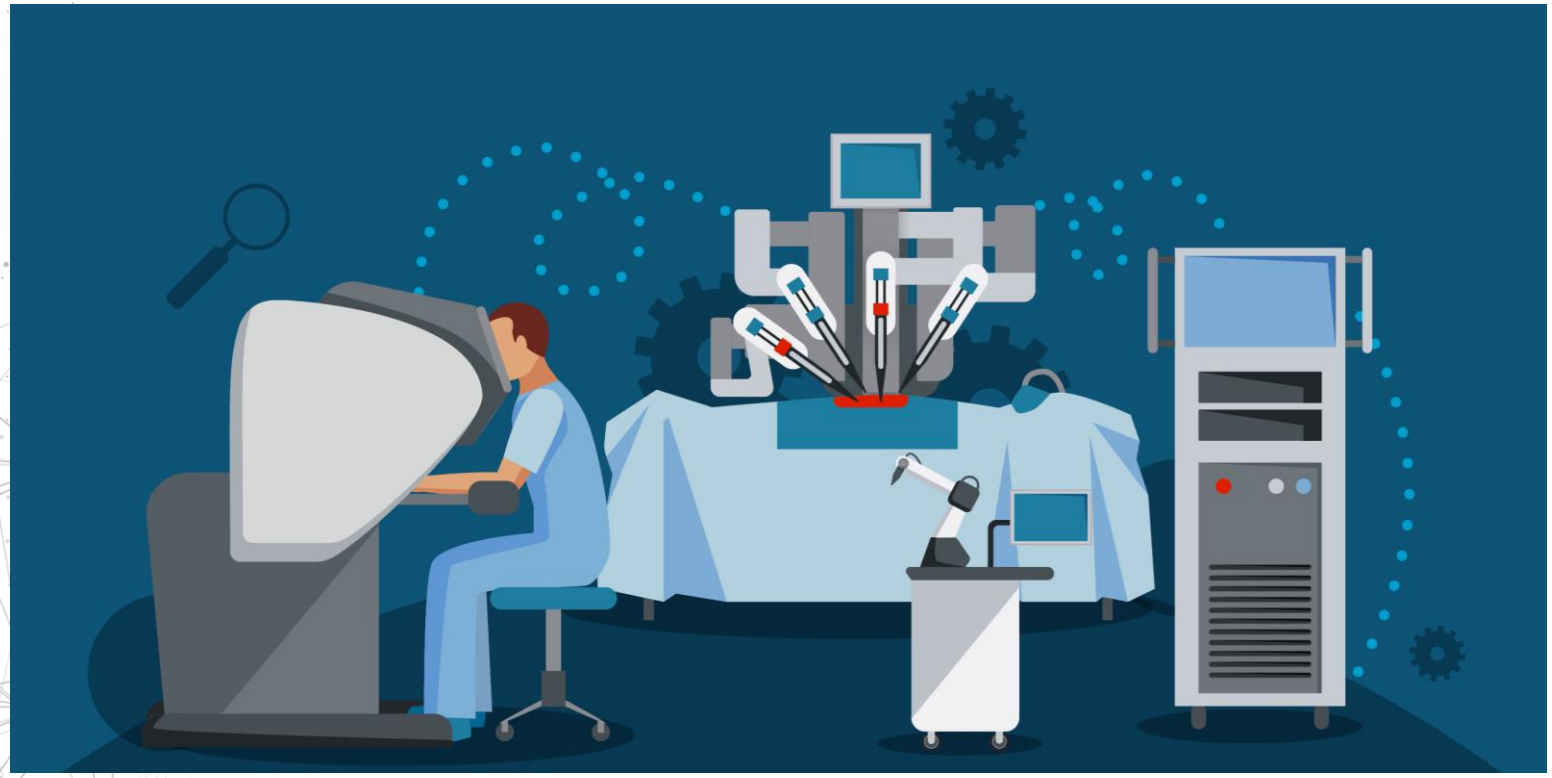
PEAS description of automated taxi?

Task environments are like problem descriptions, where rational agents are solutions!



Automated Taxi Driver
A PEAS Specification

PEAS Example
Medical diagnosis



Additional PEAS Examples

Agent Type	Performance Measure	Environmnet	Actuators	Sensors
Robot soccer playter	Winning game, goals for/against	Field, ball, own team, other team, own body	Devices (e.g., legs) for locomotion and kicking	Camera, touch sensors, accelerometers, orientation sensors, wheel/joint encoders
Internet book-shopping agent	Obtain requested/ interesting books, minimize expenditure	Internet	Follow link, enter/submit data in fields, display to user	Web pages, user requests
Autonomous Mars rover	Terrain explored and reported, samples gathered and analyzed	Launch vehicle, lander, Mars	Wheels/legs, sample collection device, radio transmitter	Camera, touch sensors, accelerometers, orientation sensors, wheel/joint encoders, radio recievers
Mathematician's theorem-proving assistant				

Task Environments Properties

Environment types .. 1

[Definitions here are informal, will be refined later in the course]

- Fully observable (vs. partially observable): An agent's sensors give it access to the complete state of the environment at each point in time.
- Deterministic (vs. non-deterministic or stochastic): The next state of the environment is completely determined by the current state and the action executed by the agent.
- *What if an agent has no sensors?*
- *Partially observable environments may appear to be stochastic*

Environment types .. 2

- Episodic (vs. sequential): The agent's experience is divided into atomic "episodes" each episode consists of the agent perceiving and then performing a single action), and *the choice of action in each episode depends only on the episode itself.*

  In the sequential case, *the next episode may depend on actions taken in previous episodes.*

Assembly line checking : episodic

Chess/Soccer agent: sequential

Environment types .. 3

- Static (vs. dynamic): The environment is unchanged while an agent is deliberating.

The environment is semi-dynamic if the environment itself does not change with the passage of time, but the agent's performance score does.

Environment types .. 4

- Discrete (vs. continuous): A limited number of distinct, clearly defined percepts and actions. (Chess vs. driving). This can describe the state of the environment, how time is handled, and the percepts + actions.

  A chess board is discrete in state environment (e.g. board configuration), discrete in time (e.g. one clock tick after each turn), and discrete in its actions as it changes from state to another.

Environment types .. 5

- Known (vs. unknown): Knowledge about outcomes for all actions given.

  What are card games?
  Known, but not necessarily fully observable.

- Single agent (vs. multi-agent): An agent operating by itself in an environment.
  Cooperative vs. Competitive multi-agents.

Environment types

Task Environment	Observable	Deterministic	Episodic	Static	Discrete	Agents
Robot soccer	Partially	Stochastic	Sequential	Dynamic	Continuous	Multi
Internet book-shopping	Partially	Deterministic*	Sequential	Static*	Discrete	Single
Autonomous Mars rover	Partially	Stochastic	Sequential	Dynamic	Continuous	Single
Mathematician's assistant	Fully	Deterministic	Sequential	Semi	Discrete	Multi

- Dimensions are independent, meaning that a discrete environment does not have to be static etc..
- The environment type largely determines the agent design
- The real world is (of course) partially observable, stochastic, sequential, dynamic, continuous, multi-agent

Structure of Agents

Agent = architecture + program

percepts from sensors → **agent program** → action choices → actuators

How do we Implement Agents

Table-driven approach, using a table of percepts and actions as in the vacuum cleaner agent?

But defining a table like this for a real system is almost impossible:
- sizes of the inputs and actions,
- effort to identify mappings between percepts and actions.

We need better types of agents,
- need to implement agent functions concisely.

Agent Types

Here's what we'll be considering in the next few slides:

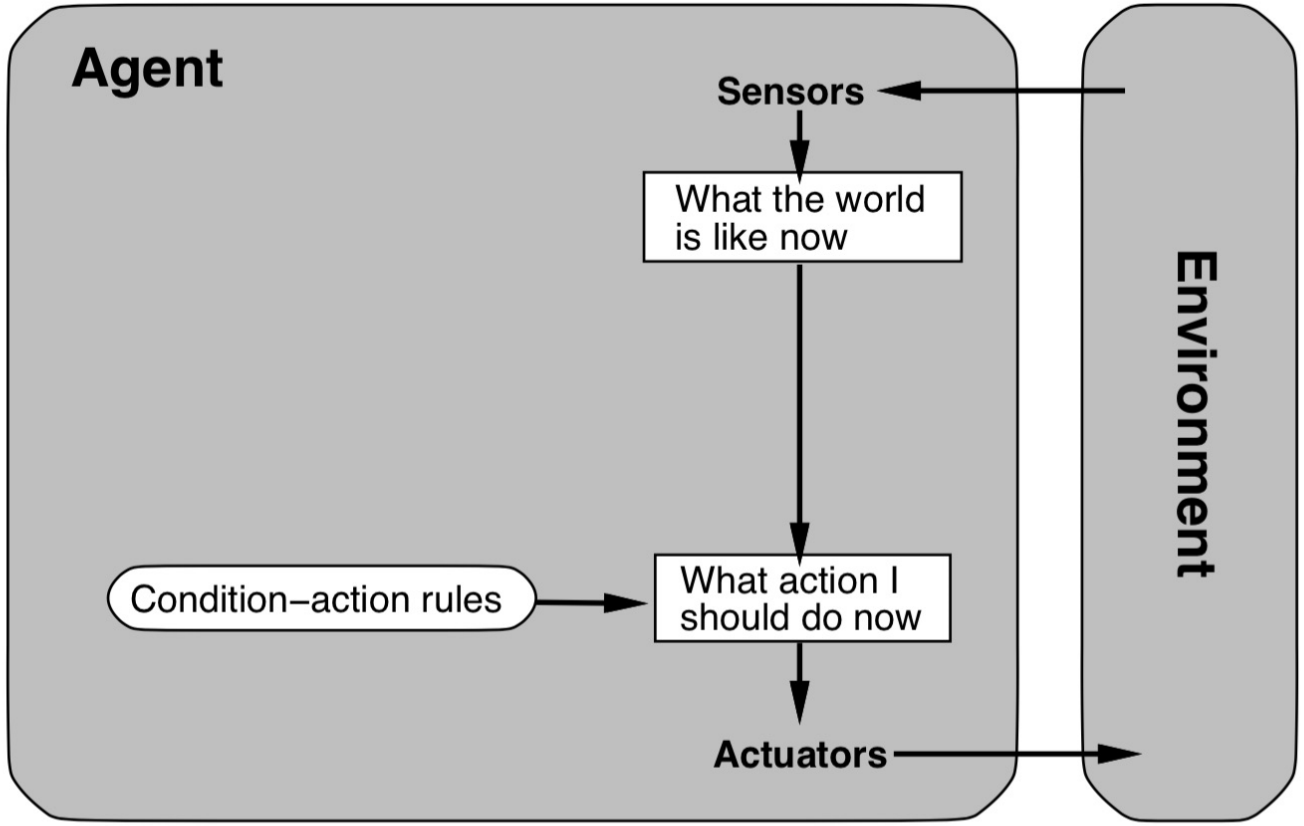
We start with the four basic agent types,
in order of increasing generality:

- Simple reflex agents
- Model-based reflex agents
- Goal-based agents
- Utility-based agents

and then we will outline:

- Learning agents

Simple Reflex Agents



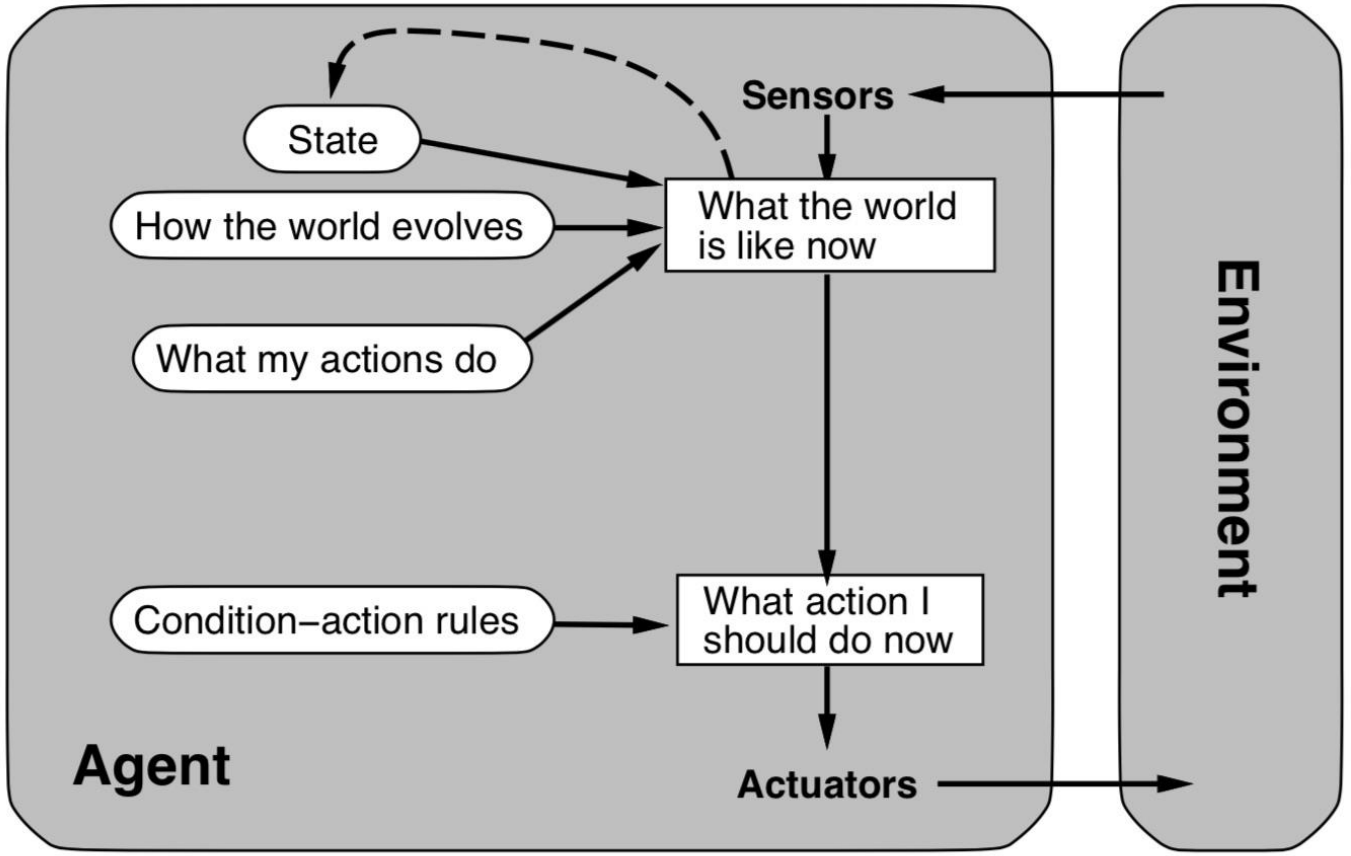
Simple Reflex Agents: Program

function SIMPLE-REFLEX-AGENT(*percept*)
returns an action
persistent: *rules,* a set of condition-action rules

state ← INTERPRET-INPUT(*percept*)
rule ← RULE-MATCH(*state, rules*)
action ← *rule*.ACTION
return *action*

Model-Based Reflex Agents

A schematic of model-based agents



Model-Based Reflex Agents: Program

function MODEL-BASED-REFLEX-AGENT(*percept*)
returns an action

persistent: *state,* agent's current conception of world state
model, a description of how the next state depends on the current state and action
rules, a set of condition-action rules
action, the most recent action, initially none

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

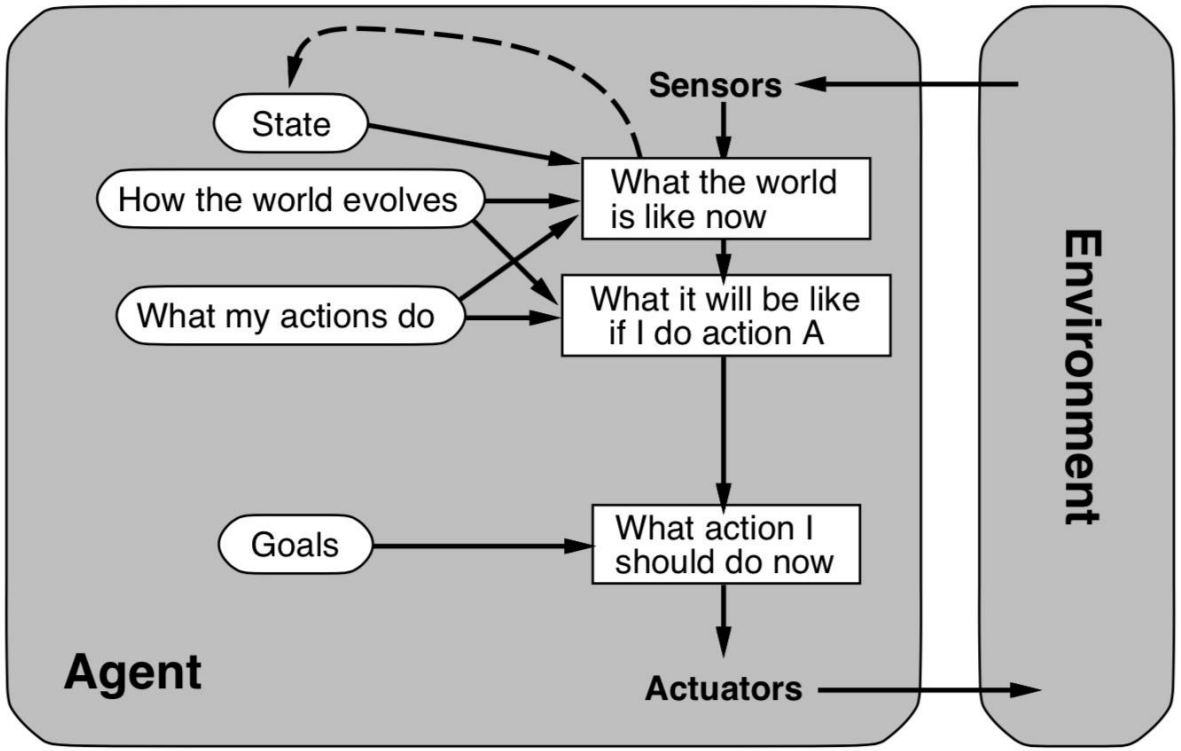
rule ← RULE-MATCH(*state, rules*)

action ← *rule*.ACTION

return *action*

Goal-Based Agents

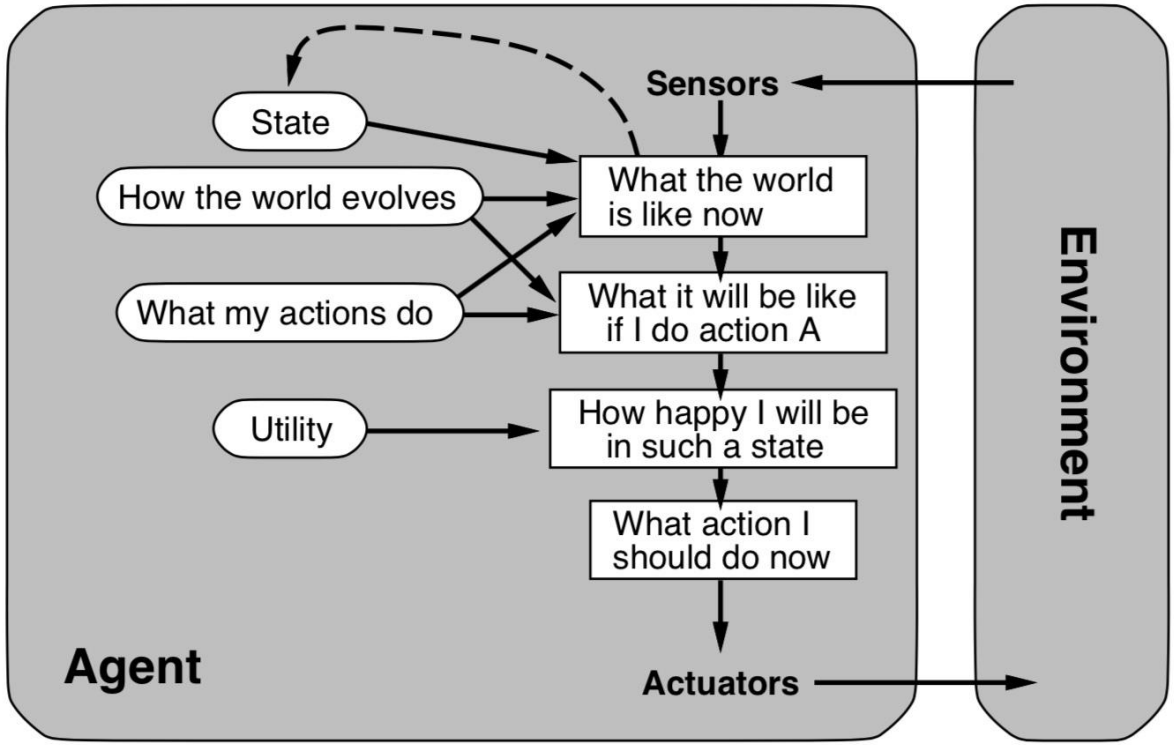
Here is the structure of model-based, goal-based agents:



- More flexible:
  - Can represent and add more knowledge to KR
  - Can change goals easily, and adapt to changing conditions.
- Can specify and get to a goal!
  - Goal may be reached in one or more actions.
    ➔ Search, Planning

Utility-Based Agents

Here is the structure of model-based, utility-based agents:

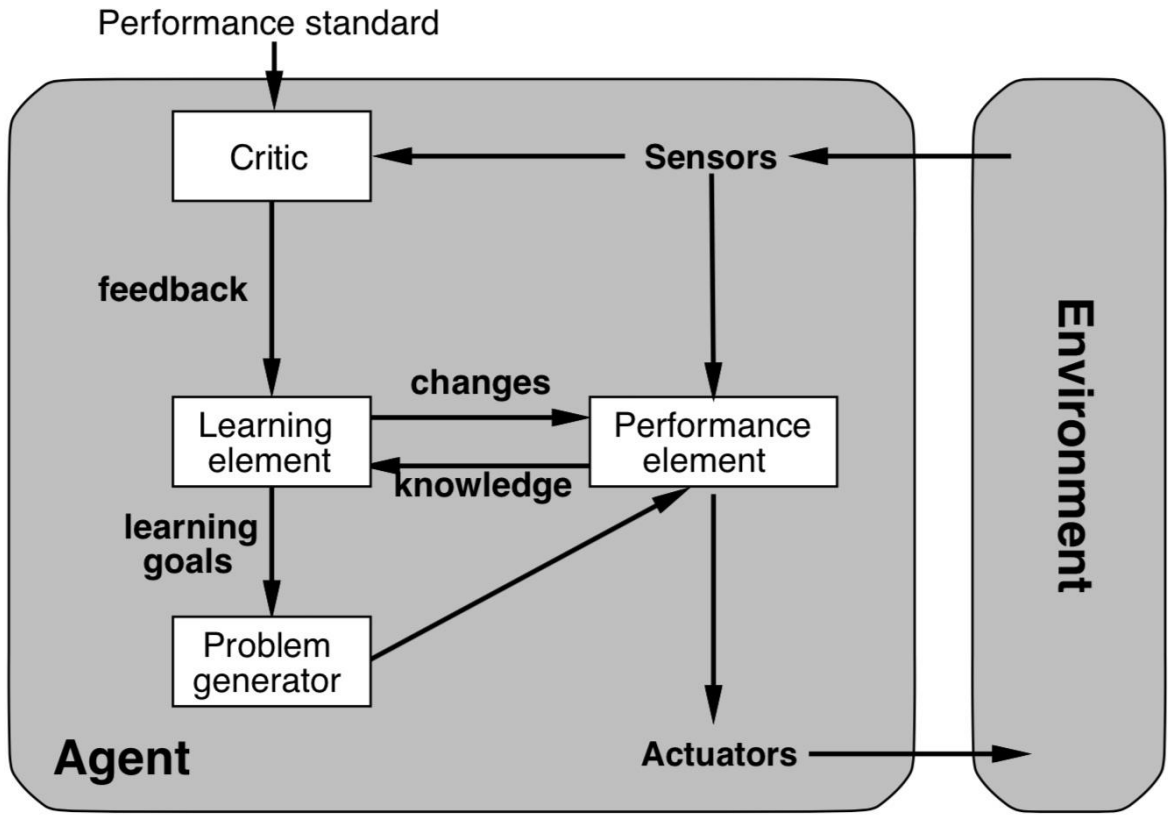


- Not just reach a goal, but do it "efficiently"/maximizing "utility"
- Not easy: many issues in modeling environment, perception, representation, reasoning, learning…
- Choosing best way to maximize utility also difficult.
  - we'll learn about some ways to do so

May need to depend on expected utility

Learning Agents

Here is the structure of a general learning agent:



- Learning: Allows agent to deal with unknown environments, become more competent.
- Learning element: how to improve; Performance element: what to do next
- Feedback from *external* critic to improve performance (e.g. coach)
- Problem generator suggests new possible actions

Environment Representations

It's useful at this point to note that there are various ways that agent environments can be represented.

- **Atomic:** a state is a black box, with no internal structure.
  E.g. In search and game-playing problems, states are atomic.
- **Factored:** a state is a vector of attribute values, where values can be Boolean, real-valued, or categorical.
  E.g. Constraint satisfaction problems, propositional logic, machine learning problems usually based on factored representations.
- **Structured:** a state includes objects, each of which has attributes of its own, as well as relationships to other objects.

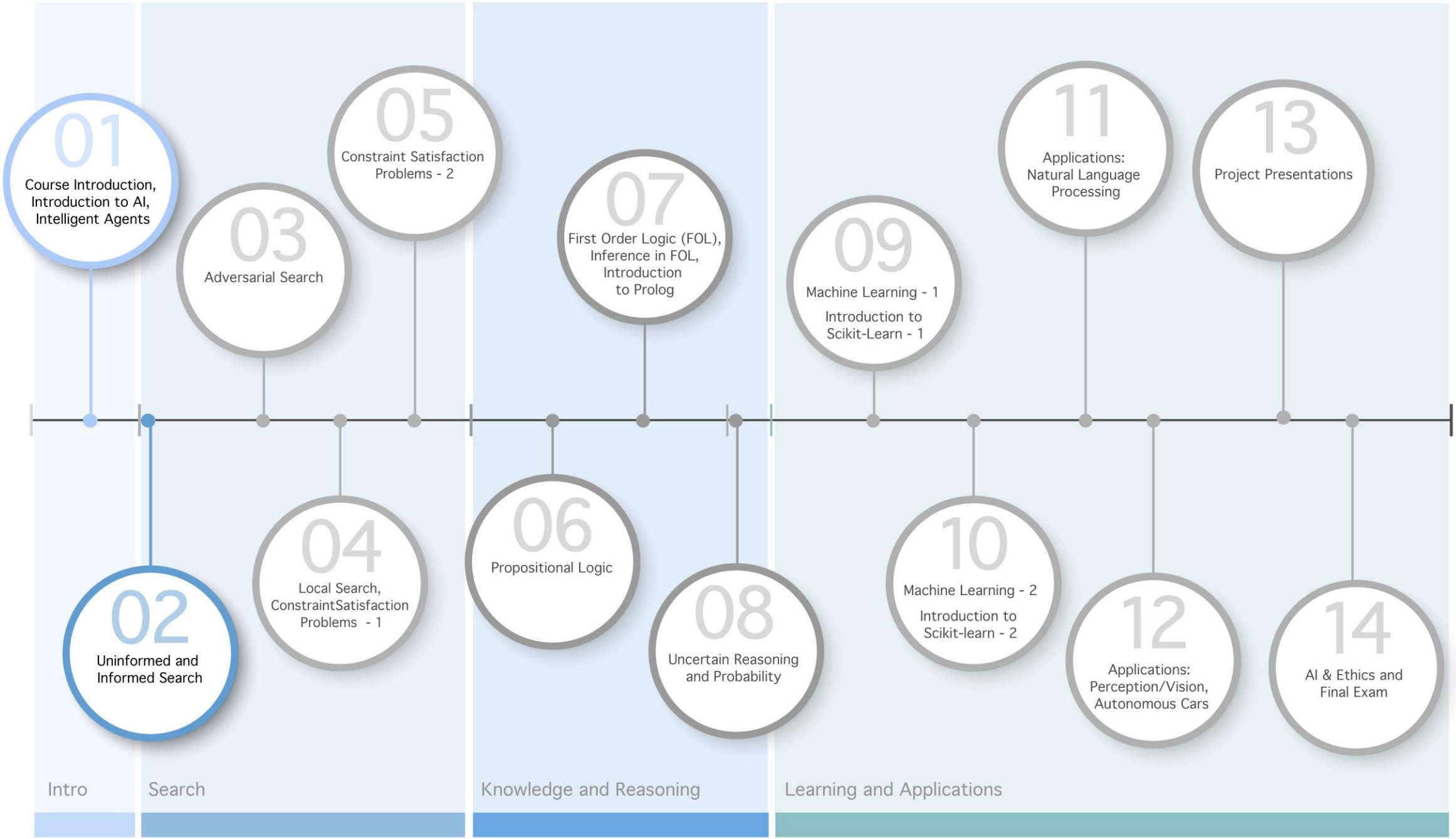
E.g. First order logic, natural language understanding, use structured representations, etc.…

Intelligent Agents: Review

- Learned about agents
- Agents interact with environments through actuators and sensors
- The agent function describes what the agent does in all circumstances
- Performance measures evaluate environment sequences
- A perfectly rational agent maximizes expected performance

- Further, agent programs implement some agent functions
- And PEAS descriptions, P-E-A-S, PEAS descriptions define task environments
- Environments are categorized along several dimensions: observable or not, deterministic or not, episodic or not, static or not, discrete or not, single-agent or not, and whether it's known or not.
- Several basic agent architectures: reflex, reflex with state information, goal based, utility based and learning

Next module: Start on search, specifically, uninformed and informed search


01
Course Introduction, Introduction to AI, Intelligent Agents
02
Uninformed and Informed Search
03
Adversarial Search
04
Local Search, ConstraintSatisfaction Problems - 1
05
Constraint Satisfaction Problems - 2
06
Propositional Logic
07
First Order Logic (FOL), Inference in FOL, Introduction to Prolog
08
Uncertain Reasoning and Probability
09
Machine Learning - 1
Introduction to Scikit-Learn - 1
10
Machine Learning - 2
Introduction to Scikit-learn - 2
11
Applications: Natural Language Processing
12
Applications: Perception/Vision, Autonomous Cars
13
Project Presentations
14
AI & Ethics and Final Exam
Intro
Search
Knowledge and Reasoning
Learning and Applications