CS2109S: Introduction to AI and Machine Learning

Lecture 1:
Intro to CS2109S and AI

14 January 2025

DO NOT CLOSE YOUR POLLEVERYWHERE APP

There will be activities ahead

Instructors

Muhammad **Rizki** Maulana

rizki@nus.edu.sg

Deep Reinforcement Learning

Conghui Hu

conghui@nus.edu.sg

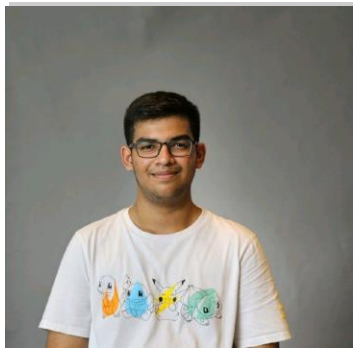
Deep Learning, Computer Vision

Patrick Rebentrost

cqtfpr@nus.edu.sg

Quantum Machine Learning

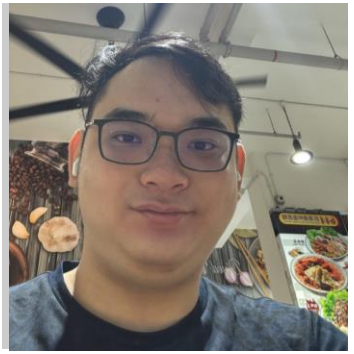
Teaching Assistant



Aprup

Nikhil Sultania

Sarji



Wilson

Ruiheng

Malaika Afra Taj

John Russell Himawan

Jalil

Gavin

Khoi Nguyen

Teaching Assistant

Chenrui Tie

Zihao XU

Ivan

Aditya

Nguyen Pham

Anton

Si Yuan

Pang Yen

Benson

Wai Kin

Teaching Assistant

Laksh

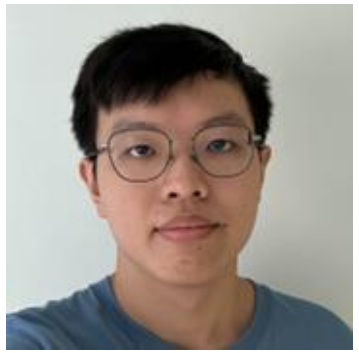
Chun Jie

Thanh Chu

Daryl

Ian

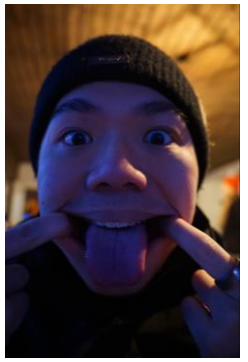
Jiacheng



Zheng Long

Diwen

Yi Hong



Phi

Teaching Assistant

Shaun Quek

Kum Chai Yin

Wei Hao

Rights Infringements on NUS Course Materials

All course participants (including permitted guest students) who have access to the course materials on Canvas or any approved platforms by NUS for delivery of NUS modules are not allowed to re-distribute the contents in any forms to third parties without the explicit consent from the module instructors or authorized NUS officials

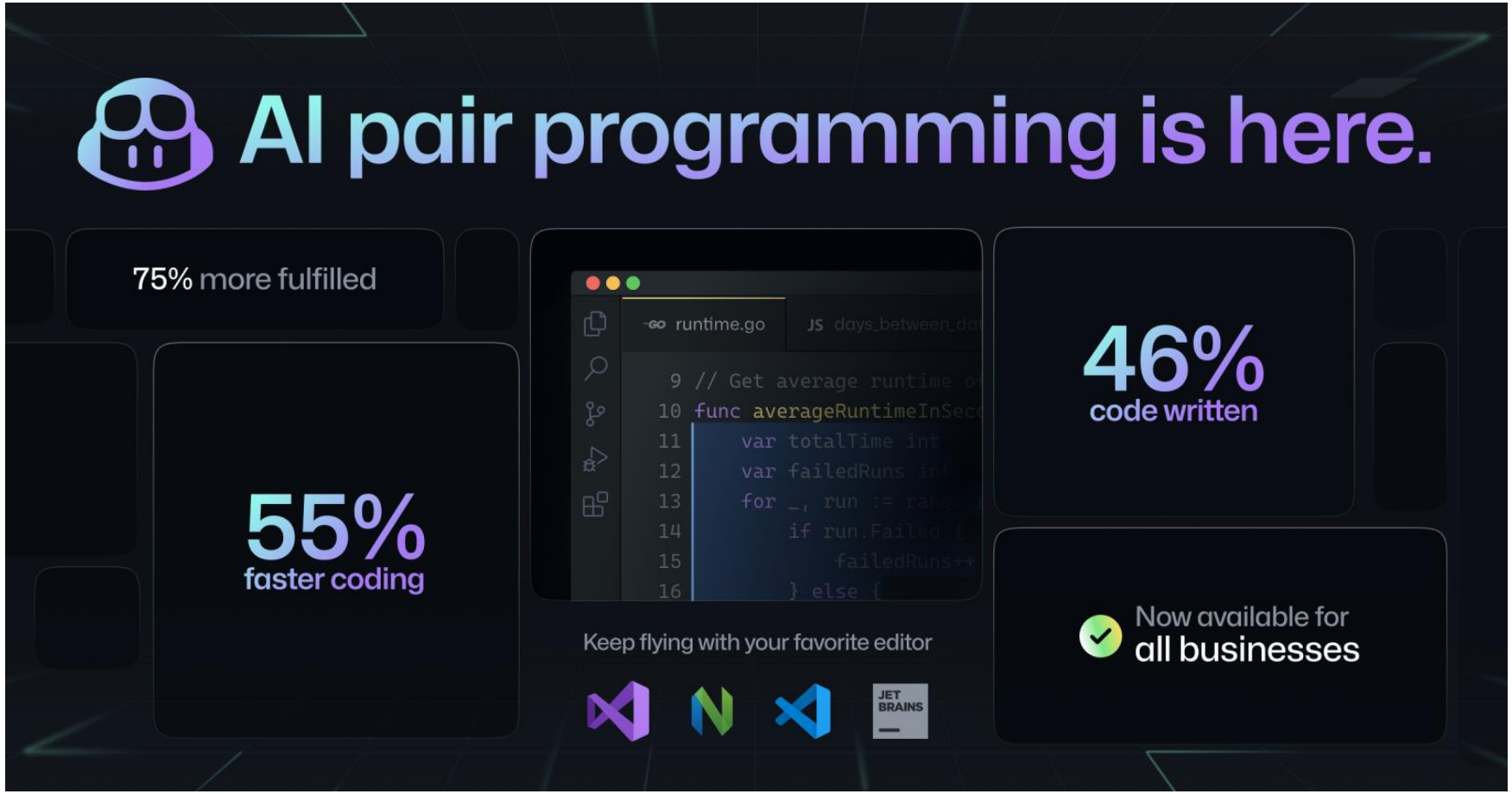
Why do we want to learn about AI & ML?


NEWS|EXCLUSIVE
SAM ALTMAN
CEO OF OPENAI
abc NEWS

Wouldn’t you want to understand how these work?

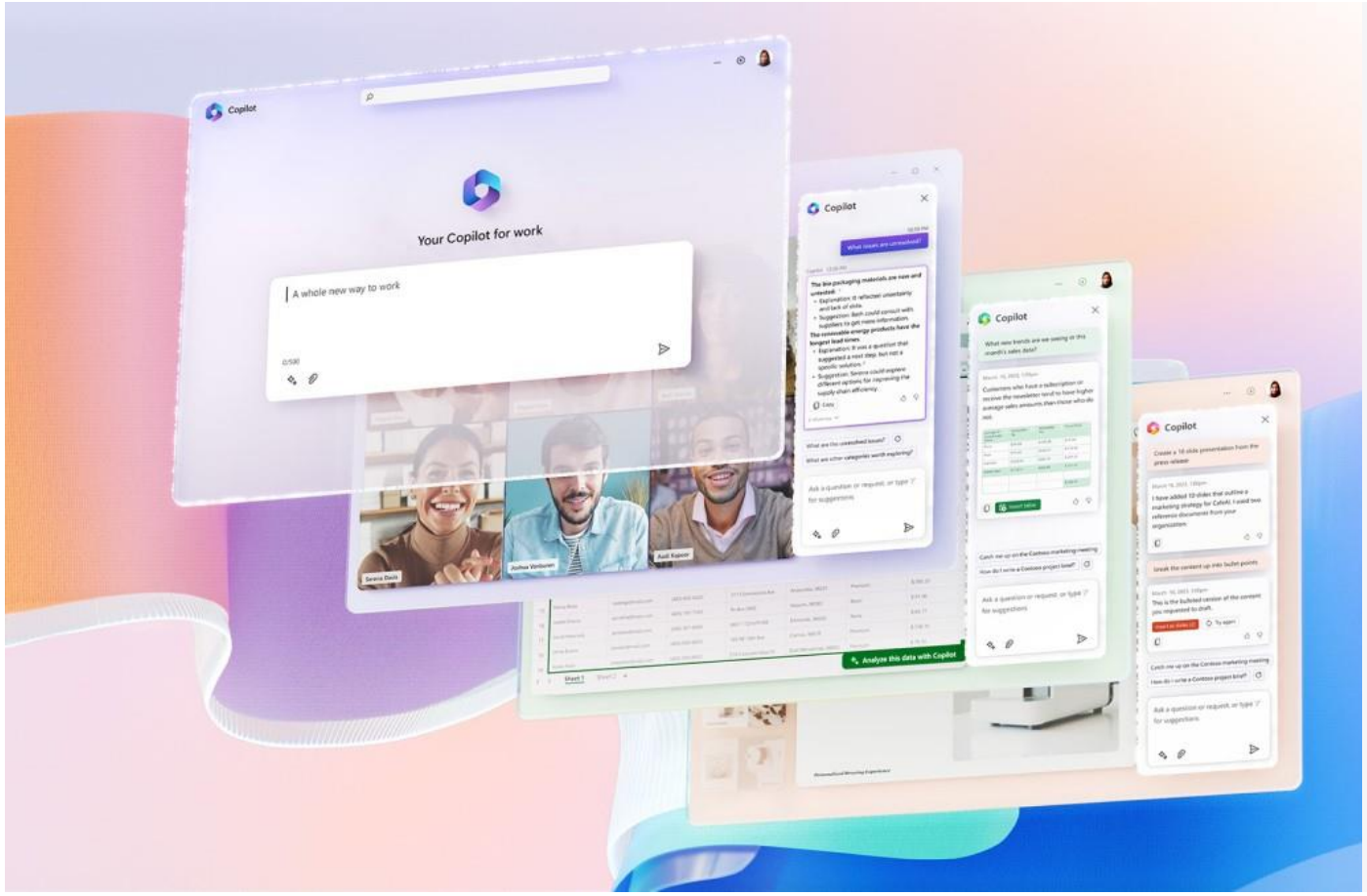
This course is the first step

Software Development with AI

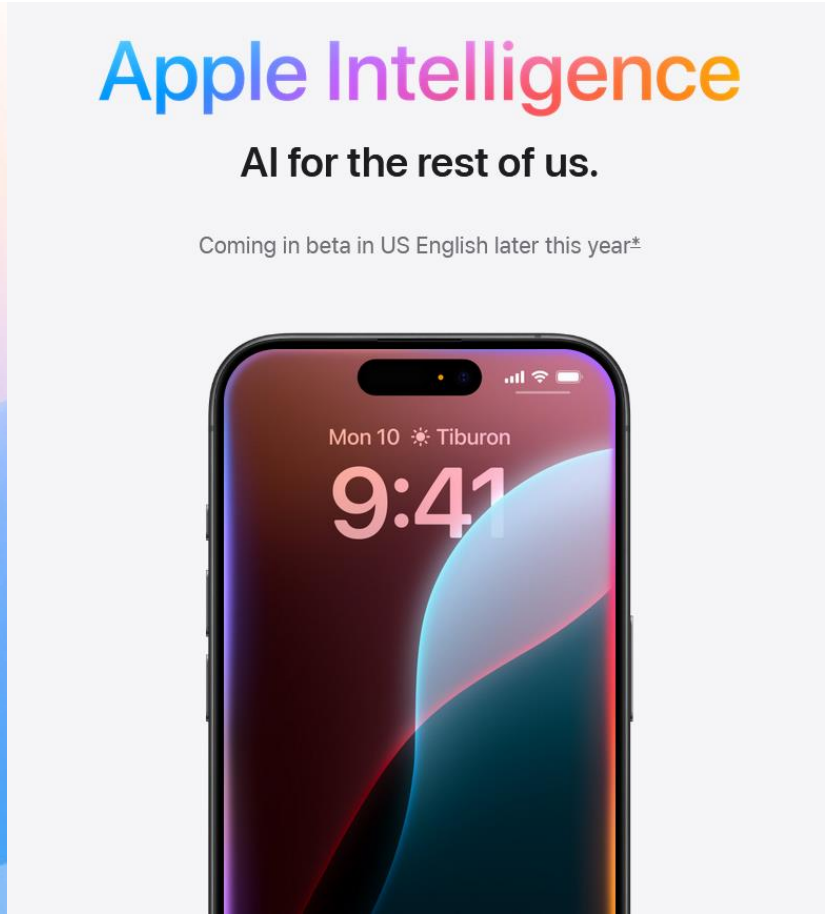


Credit: GitHub

Companies adding AI into their products

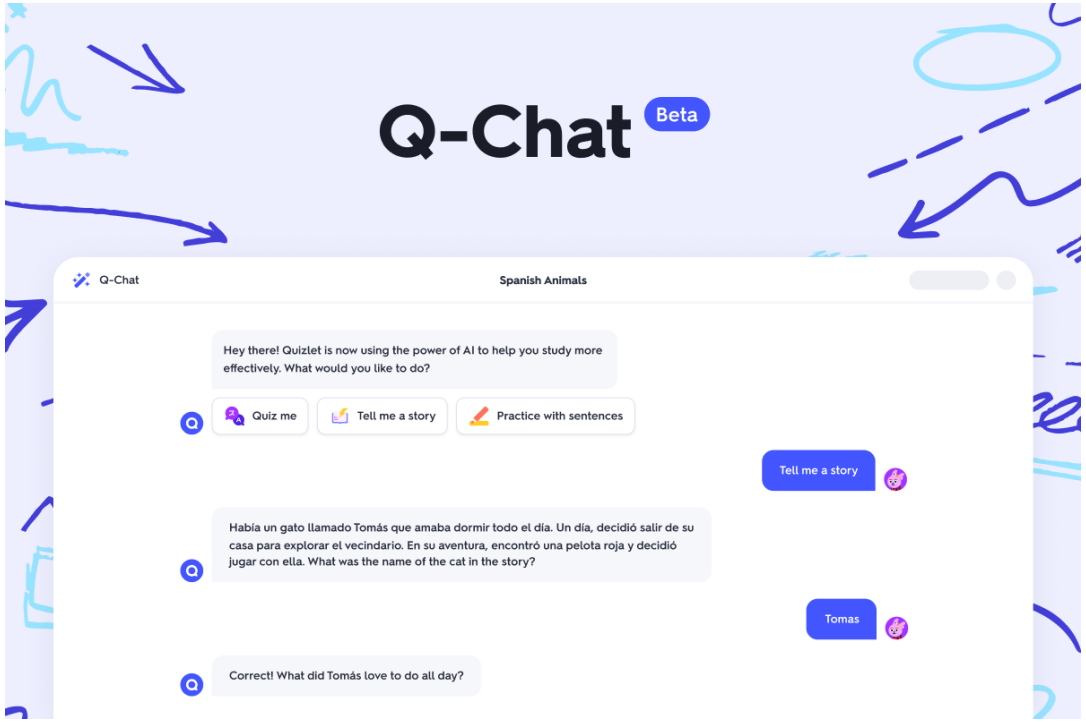


Credit: Microsoft



Credit: Apple

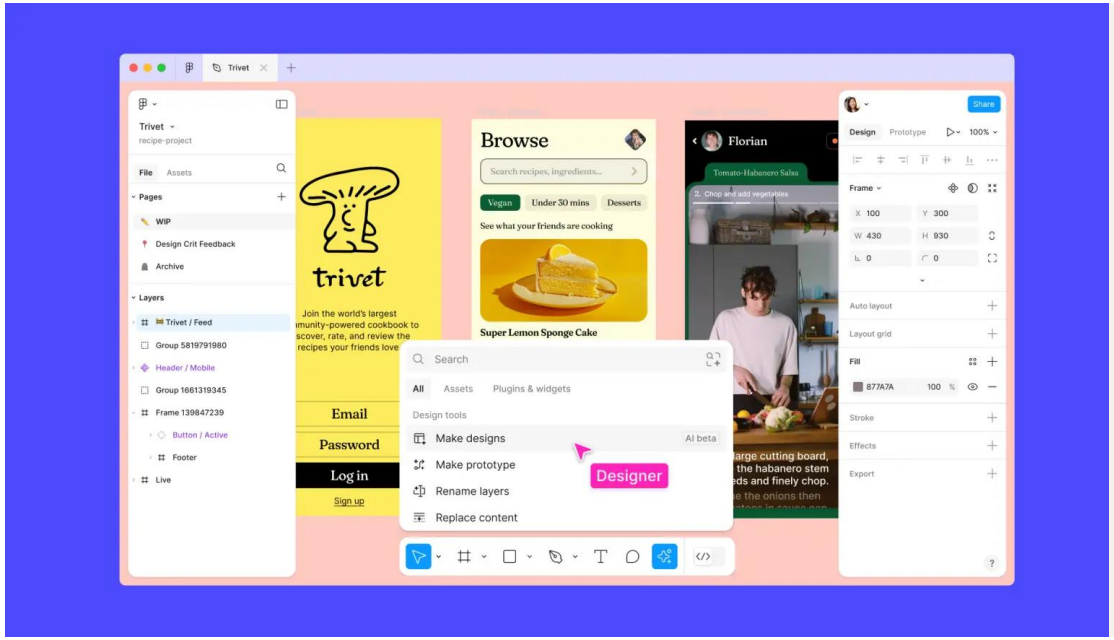
Companies adding AI into their products



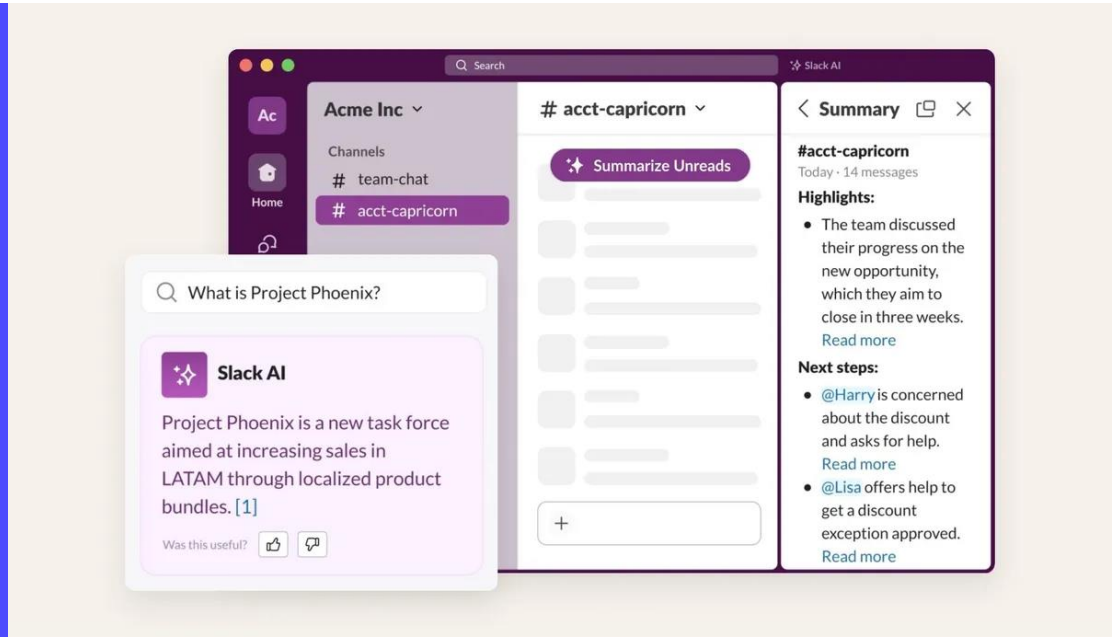
Credit: Quizzlet

Credit: sitthiphong / Shutterstock.com

Companies adding AI into their products



Credit: Figma AI



Credit: CNET

Companies expect software engineers to know about AI

Learning the underlying design principles

Understand ideas, concepts, and techniques used for AI:

What are some of them?

How to use these?

What are the limitations?

By learning AI & ML, we learn how machines learn

Learning how to learn!

Intro to CS2109S

CS2109S

An introductory course designed to cover the fundamental topics in AI and ML.

What to Expect:

- A wide range of introductory topics in "classical" AI and ML.
- Practical knowledge on implementing basic AI and ML algorithms and applying them to solve problems.
- Conceptual understanding of key principles and foundational concepts in AI and ML.

What Not to Expect:

- A course focused on theoretical depth with rigorous proofs, mathematics, and theorems.
- Extensive coverage of the latest advancements in AI and ML.

These are features, not bugs.

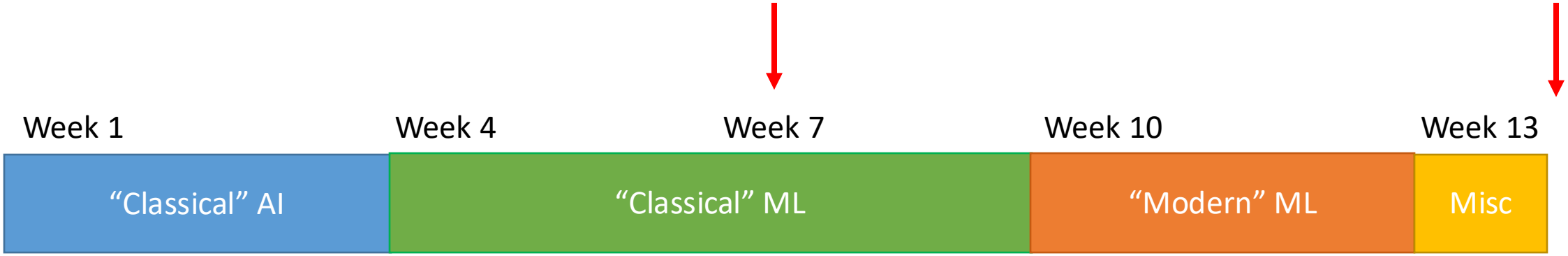
Course Pre-requisites

- CS1101S, CS1010S or equivalent
- CS1231 or equivalent
  - Trees, graphs, counting & combinatorics
- MA1521
  - Differentiation, chain rule
- CS2040S or equivalent
  - Tree and graph search
- Linear algebra
  - Vector, matrix, and their operations
- Python

Overview

Midterm
(During lecture time L1)

Final Exam
(Exam Week)



Search Algorithms

- Uninformed search: BFS, DFS
- Local Search: Hill Climbing
- Informed search: A*
- Adversarial search: Minimax

"Classical" ML

- Decision Trees
- Linear/Logistic Regression
- Kernels and Support Vector Machines
- "Classical" Unsupervised Learning

"Modern" ML

- Neural Networks
- Deep Learning
- Sequential Data

Miscellaneous

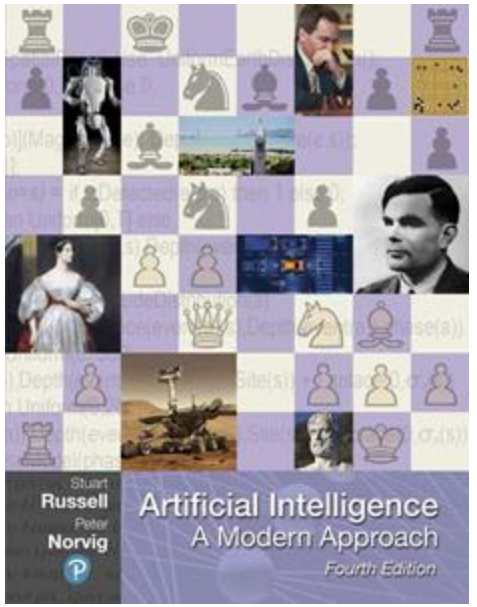
- AI & Ethics

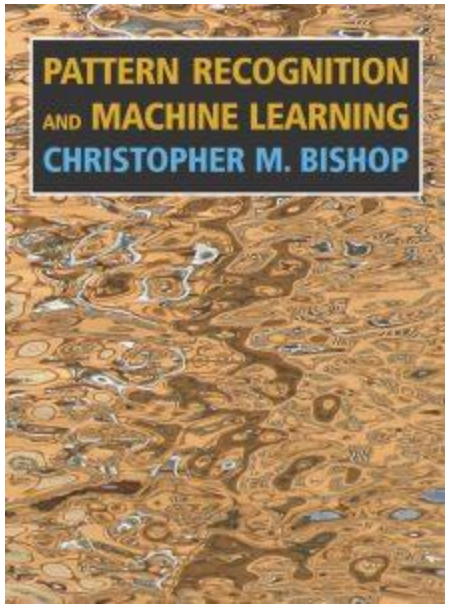
Applied CS2040S, CS1231
Python

Applied Linear Algebra, Calculus, Statistics & Probabilities
Numpy, Scikit-learn, PyTorch

Textbooks

- Russell and Norvig (2021) Artificial Intelligence: A Modern Approach (4th Ed)
- Christopher Bishop (2006) Pattern Recognition and Machine Learning





There is no need to buy these books.

Learning Management System

Only for **Gradebook** to record marks
for checking and **webcast**

Lecture Slides

Lecture slides will be released **after** the lecture.

Why?

- We are preparing the slides before the lecture
- Swiping through the slides to catch up or preview ahead can distract you from following the lecture

Collaboration Policy

- Collaborate on assessments **only** to the extent of **discussing ideas**, not (potential) implementation or (potential) code. All discussions must be **cited**.
- It is okay to incorporate a few lines of code that you find online or elsewhere into your own code, **provided that those lines are not themselves solutions** to assigned work and that you **cite** the lines' origins.
- Avoid sharing, consulting, discussing, or using (parts of) solution codes or their essence **through any means**, including chat, spoken words, whiteboard, video, or brief glances. This sharing policy also applies after you have completed the module.
- Do not view or use another person's solution as the basis for your own solution.

If you have any doubts about whether your actions are permissible, please contact us.

Source: https://canvas.nus.edu.sg/courses/69929/pages/course-policies

AI Policy

The policy regarding the use of AI varies between courses. For this course:

- Using AI is permitted as long as proper citations are provided.
  - If it is possible to share a link to the conversation (e.g., ChatGPT, Bing), then provide the link; if not (e.g., GitHub Copilot), then take screenshots
  - Document any changes made to the code generated by AI
- It is advisable to use AI-generated code as it is; alteration/obfuscation is strongly discouraged.

Source: https://canvas.nus.edu.sg/courses/69929/pages/course-policies

Citation Policy

The motivation for this citation policy is that you clearly identify the origins of each part of your submission.

- All discussions, few lines of code from online sources, and AI conversations **must** be cited **at the point in the code** (e.g., beginning of a code block) where they have influenced your code **via a python comment**.
- For **conceptual (non-code) questions,** provide the appropriate citations in the respective answer field. You may use the # character to indicate a comment.
- Provide any screenshots in the "Reference" section.

Source: https://canvas.nus.edu.sg/courses/69929/pages/course-policies

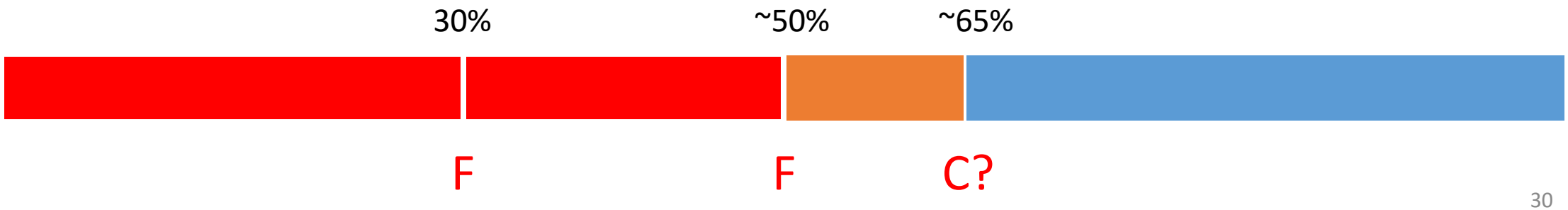
On Plagiarism

- Any plagiarism or academic dishonesty will be reported to UG.
- Plagiarism checker will be performed against all previous batches!
- Potential penalty (source: UG office)
  - $\geq 10\%$ - FAIL
  - $< 10\%$ - max: reduction in letter grade

Source: https://canvas.nus.edu.sg/courses/69929/pages/course-policies

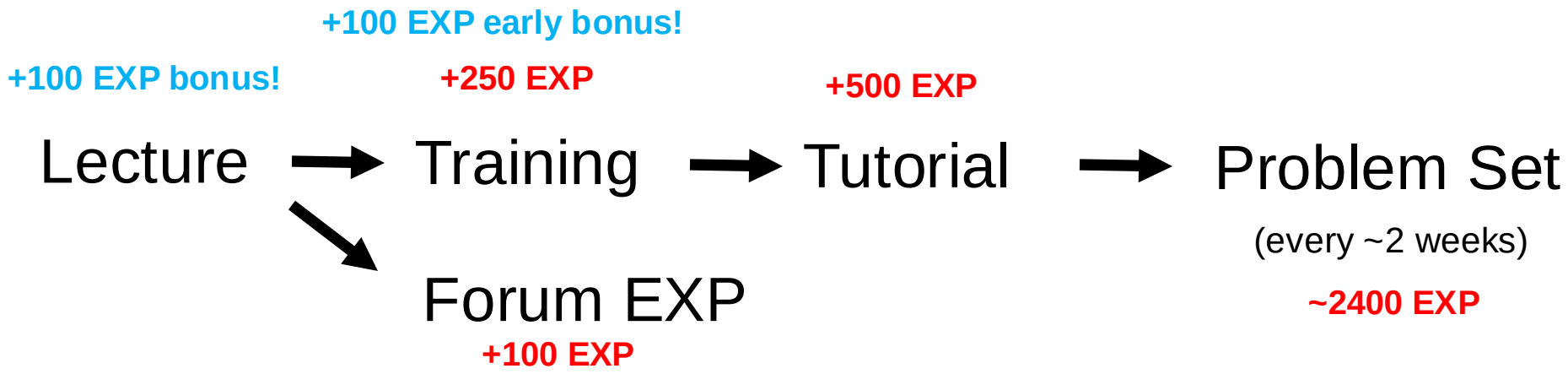
Assessments

Name	Percentage
Coursemology (CA)	30%
Mini-Project	10%
Midterm	30%
Final Exam	30%



Gamified CA (30%)

Formative



- Enough EXP? Level Up!
- **Your <u>final level</u> is your CA grade**
  - Capped at 30

Background Survey

+100 EXP

Gamified CA (30%)

Formative

- Lecture
  - 100 bonus EXP if you participate in your lecture slot (i.e., answer questions)
- Tutorial
  - You'll be given a participation EXP by your TA
  - EXP spectrum:
    - 300 (baseline default score, attend with minimal participation),
    - 400 (active participation),
    - 500 (outstanding)

Mini-Project (10%)

Exploration

Develop an AI agent to solve a problem (more details later!).

- You can use any techniques covered in the course, including "classical" AI, "classical" ML, or "modern" ML, or explore approaches beyond the course material.
- The AI agent will **be compared against our agents**. Grades will be based on your agent's performance.
- The top-performing agents will be showcased in the final lecture, with **prizes** awarded to the winners!

Duration: ~2 months

Midterm and Final

Summative

Closed book, open-sheet

Focus on application, not memorization

- **Midterm (30%)**
  - Date/Time: Tuesday, 4 March (Week 7), 4.15pm - 5.45pm
  - Venue: MPSH 2A & 2B
- **Final (30%)**
  - Date/Time: Monday, 5 May (Exam Week), 9.00am - 11.00am
  - Venue: TBA

Late Policy (Problem sets)

Penalties:

- Up to 1 hour, 0%
- Up to 24 hours, 20%
- Up to 3 days, 30%
- Beyond 3 days, 50%

If you need an extension (for valid reasons), please ask early

Common Mistakes (Problem sets)

- **Debugging algorithms with Jupyter Notebook**
  - Should use IDE (e.g., Visual Studio Code, IDLE) for PS0 and all other non-ML problem sets
- **Not reading documentations and reinventing the wheels**
  - There are many helpful functions built-in
- **Coding directly on Coursemology**
  - Should code and test locally

Intro to Artificial Intelligence

Outline

- What is AI?
- A Brief History of AI
- Intelligent Agents
  - PEAS Framework
  - Properties of Task Environment
  - Structure of Agents
  - Exploration vs Exploitation

Outline

- **What is AI?**
- A Brief History of AI
- Intelligent Agents
  - PEAS Framework
  - Properties of Task Environment
  - Structure of Agents
  - Exploration vs Exploitation

Examples of AI?

ChatGPT

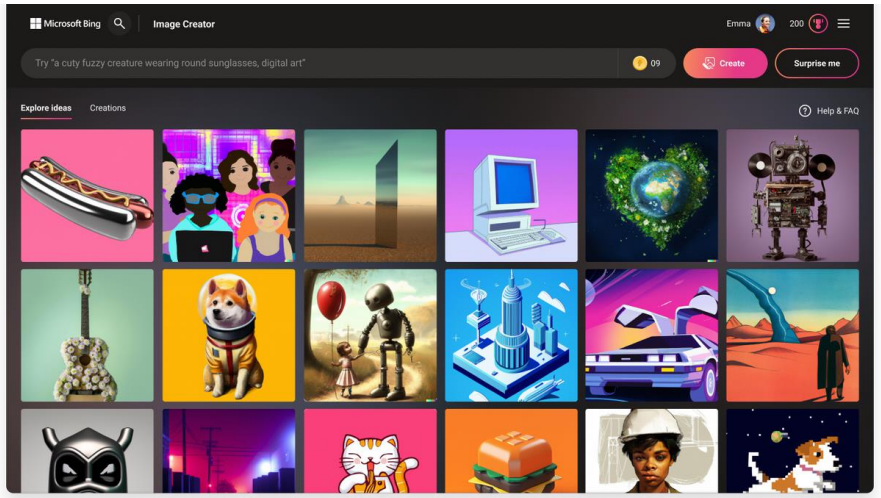
Credit: IEEE Spectrum

Credit: Guardian

Credit: NYTimes

Examples of AI?

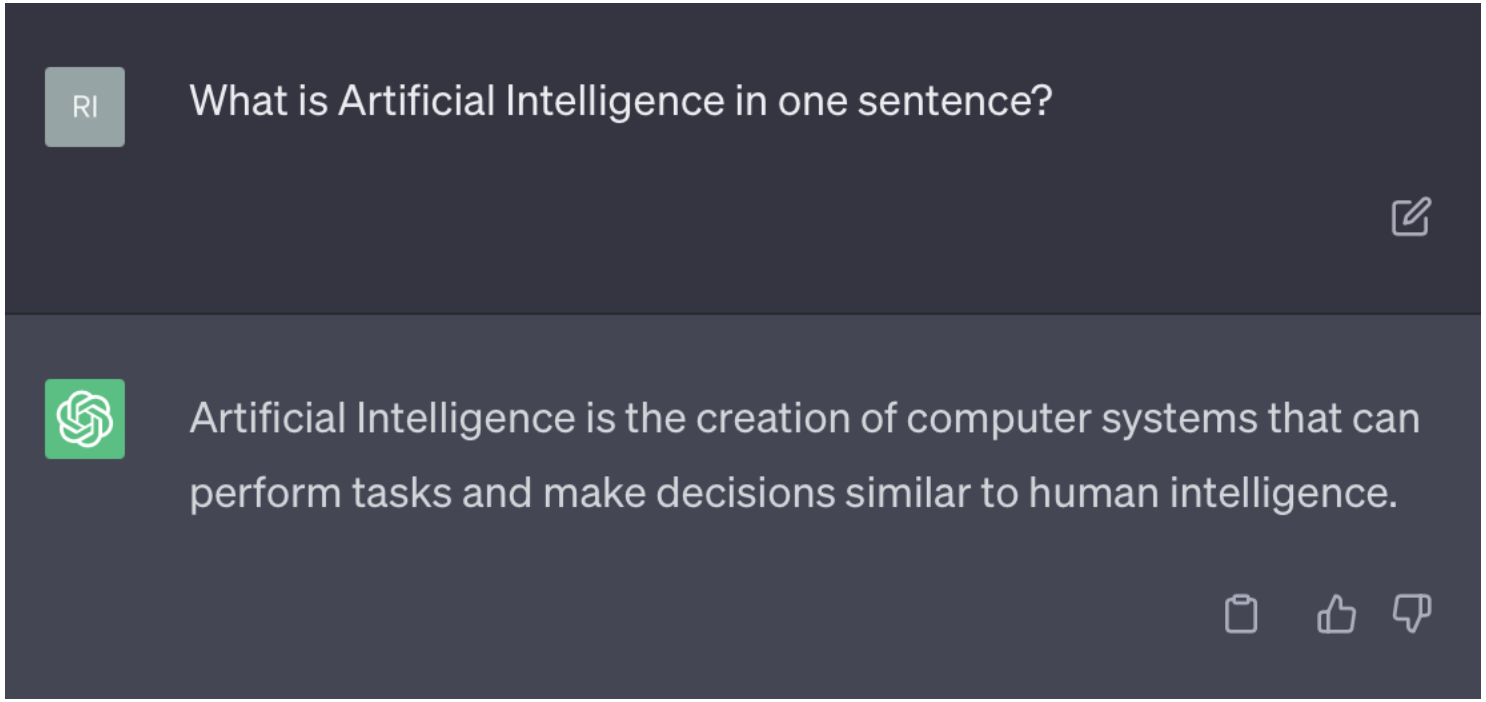
Credit: Tesla



Credit: Microsoft

Credit: Eden AI

What is AI? Let's ask AI



What is AI?

“We define AI as the study of agents that receive percepts from the environment and perform actions.”

- Russel and Norvig, AIAMA 2nd edition

“The science and engineering of making intelligent machines”.

- John McCarthy, 1955

Outline

- What is AI?
- **A Brief History of AI**
- Intelligent Agents
  - PEAS Framework
  - Properties of Task Environment
  - Structure of Agents
  - Exploration vs Exploitation

A Brief History of AI

The beginning of the universe

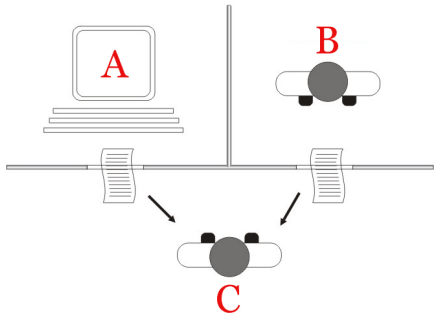
- No AI research

1900s

- Atanasoff-Berry Computer: solve linear equations
- Artificial neurons

1950s

- Alan Turing: Turing Machines, Turing Test
- Checkers AI
- Lisp programming language
- ELIZA



1970s

- 1st AI Winter

1980s

- Expert Systems
- Fifth-generation Computers

1990s

- Deep Blue

2000s

- 2nd AI Winter
- Internet & Big Data
- DARPA Grand Challenge

Credit: Wired

2010s: Deep Learning

- Deep neural networks
- IBM Watson
- Apple's Siri, Google Now, Cortana

Credit: Yonomi

2015s: Deep Reinforcement Learning

- AlphaGo, AlphaStar, OpenAI Five
- Transformers, GPT

Credit: Guardian

2020s: Generative AI

- ChatGPT
- Dall-E, Stable-diffusion, Sora
- Multimodal generative AI
- AI in consumer products

Break

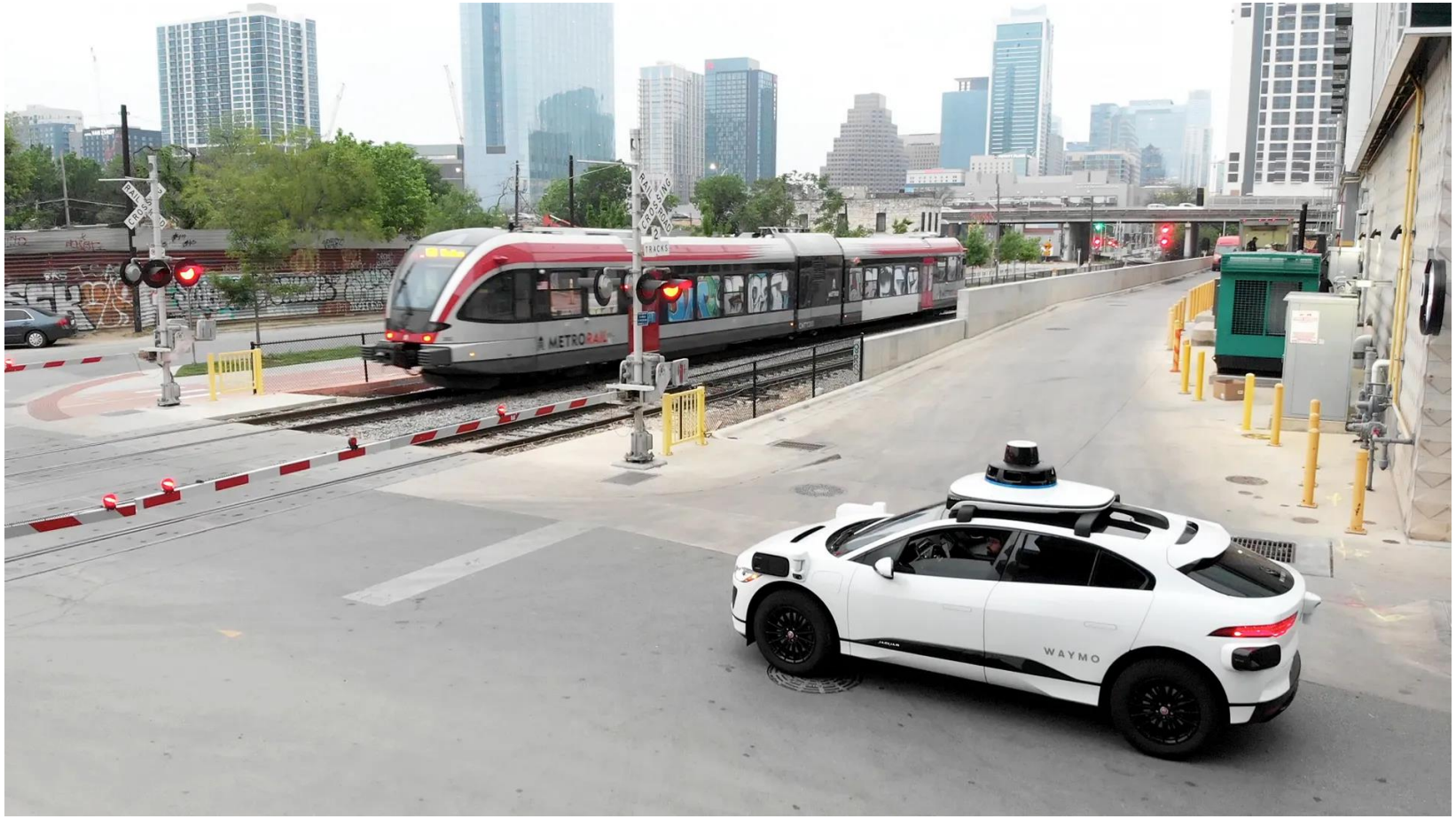

THINGS ARE GOING TO GET WEIRD
TEDEd

Outline

- What is AI?
- A Brief History of AI
- **Intelligent Agents**
  - PEAS Framework
  - Properties of Task Environment
  - Structure of Agents
  - Exploration vs Exploitation

Intelligent Agents

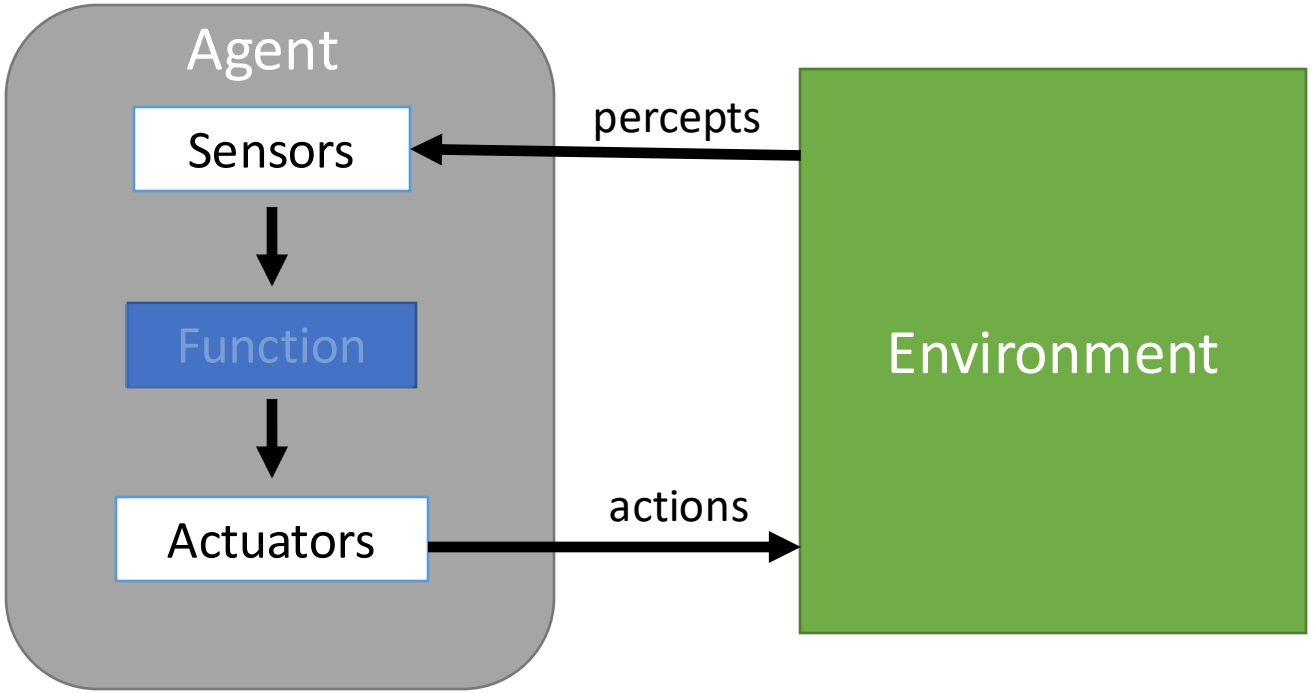
How to design an intelligent agent?



Credit: Axios

Intelligent Agents

Performance Measure, **E**nvironment, **A**ctuators, **S**ensors



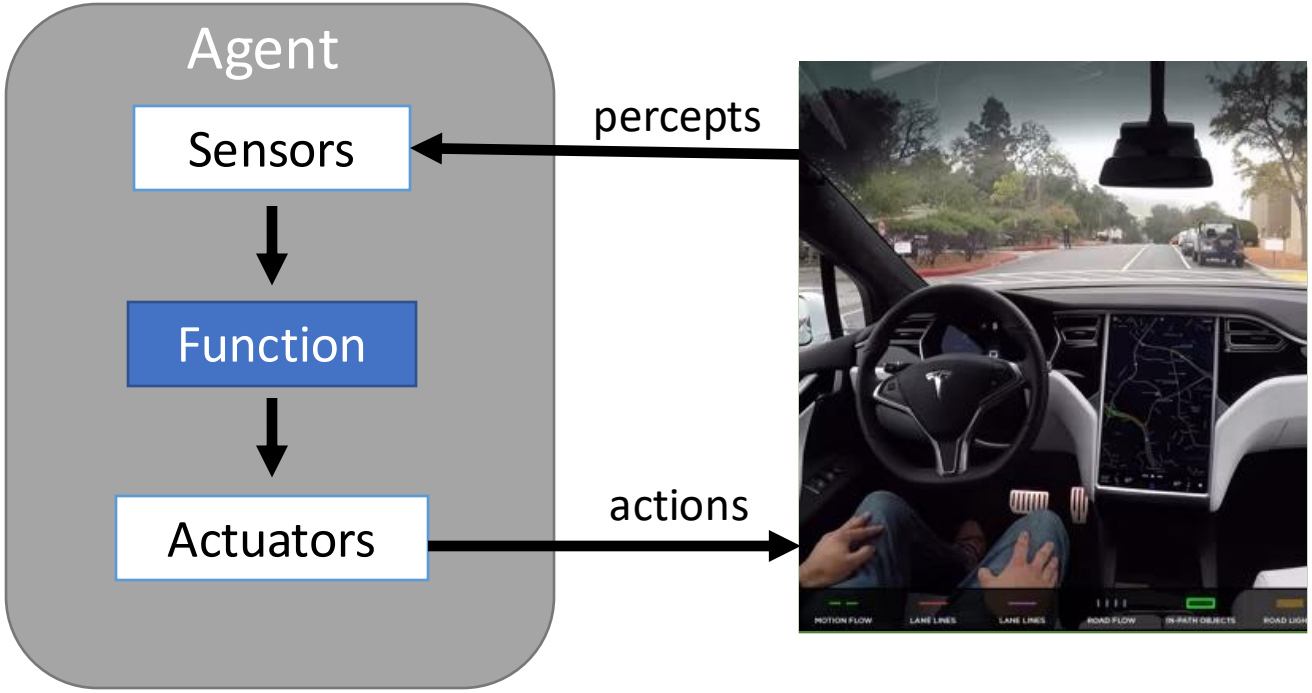
Performance Measure

Things to consider:

- Best for whom?
- What are we optimizing?
- What information is available?
- Any unintended effects?
- What are the costs?

A **rational agent** will choose actions that maximize performance measure.

Intelligent Agents: Self-Driving Car



Performance Measure, **E**nvironment, **A**ctuators, **S**ensors

Intelligent Agents: AI Assistant/Chatbot

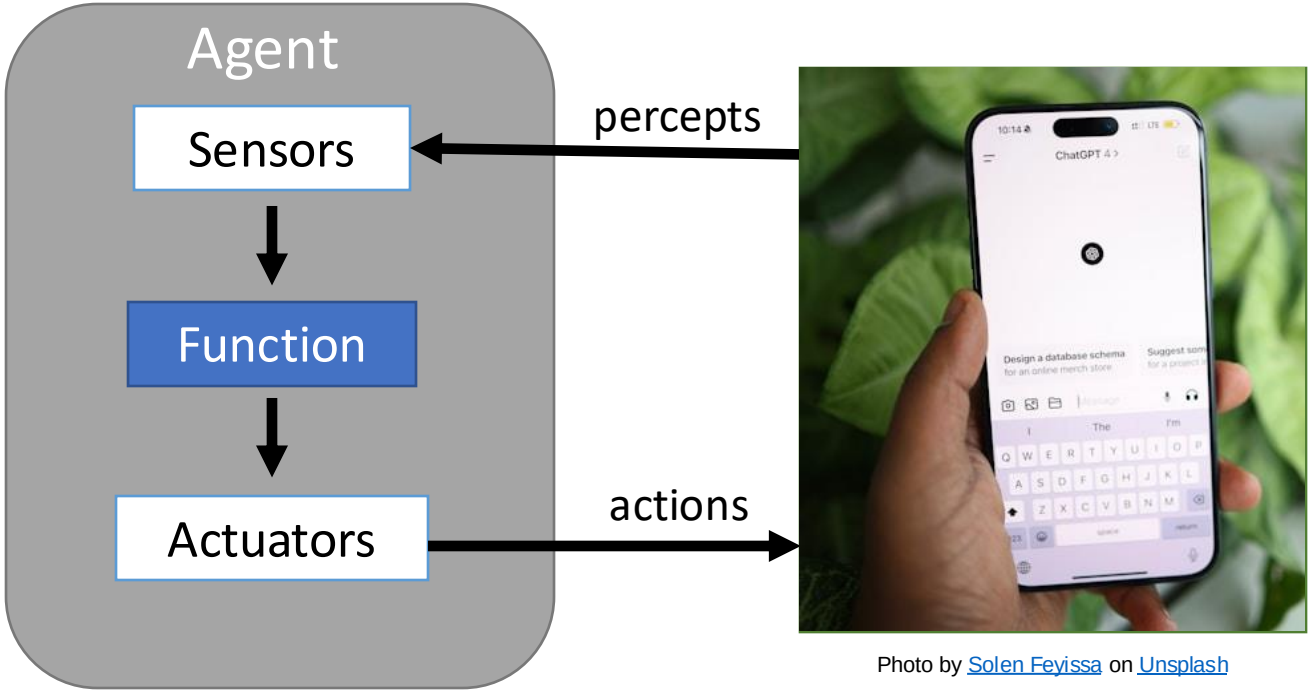
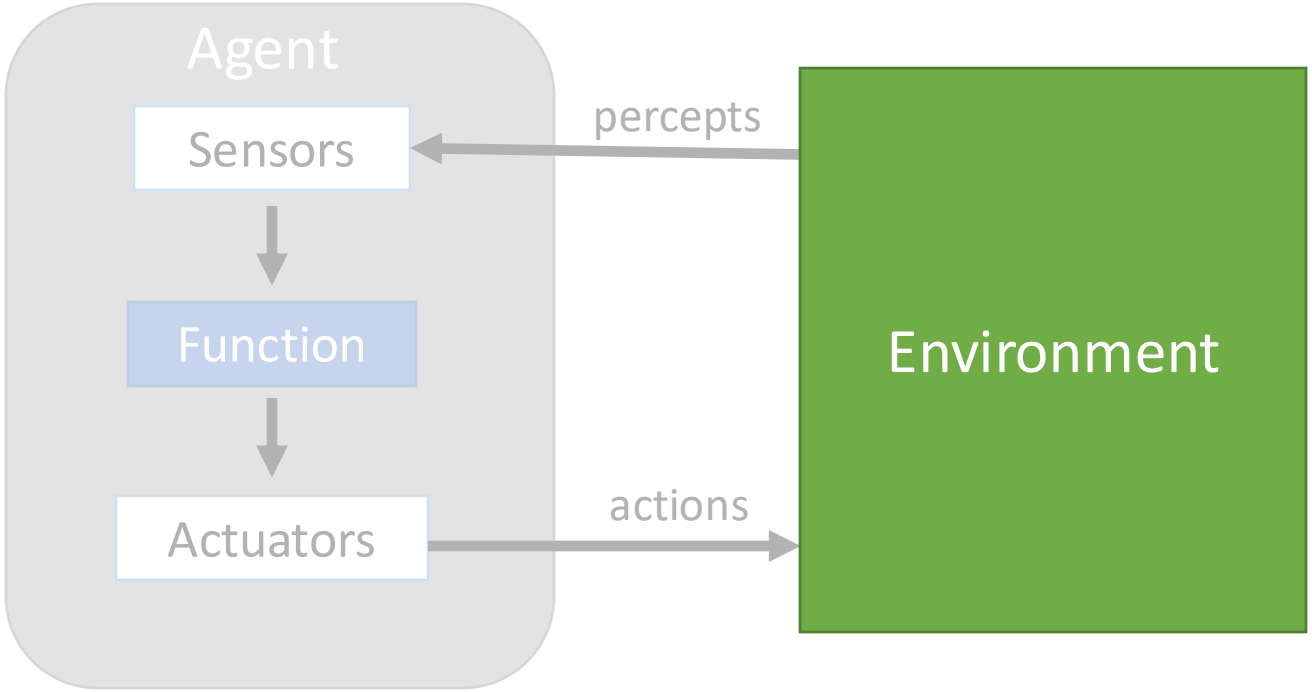


Photo by Solen Feyissa on Unsplash

Performance Measure, **E**nvironment, **A**ctuators, **S**ensors

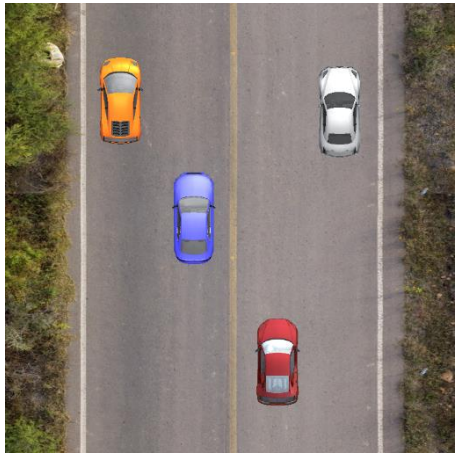
Intelligent Agents



Properties of Task Environment

Full

Partial



Credit: Google Play

Partial

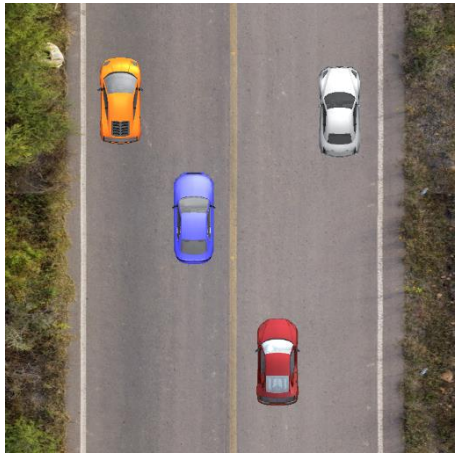
Fully observable (vs. partially observable)

An agent's sensors give it access to the complete state of the environment at each point in time.

Properties of Task Environment

Deterministic, Strategic

Stochastic, Strategic



Credit: Google Play

Deterministic

Deterministic (vs. stochastic)

The next state of the environment is completely determined by the current state and the action executed by the agent.

If the environment is also dependent on the actions of other agents, then it is also **strategic**, unless the other agents are predictable (e.g., "unintelligent").

Properties of Task Environment

Sequential

Sequential

Credit: Keyence

Episodic

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.

Properties of Task Environment

Static/Semi-Dynamic

Credit: Keyence

Static/Dynamic

Dynamic

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)

Properties of Task Environment

Discrete

Continuous

Credit: Keyence

Continuous

Discrete (vs. continuous)

A limited number of distinct, clearly defined percepts and actions.

Properties of Task Environment

Multi-agent

Multi-agent

Credit: Keyence

Single-agent

Single agent (vs. multi-agent)

An agent operating by itself in an environment.

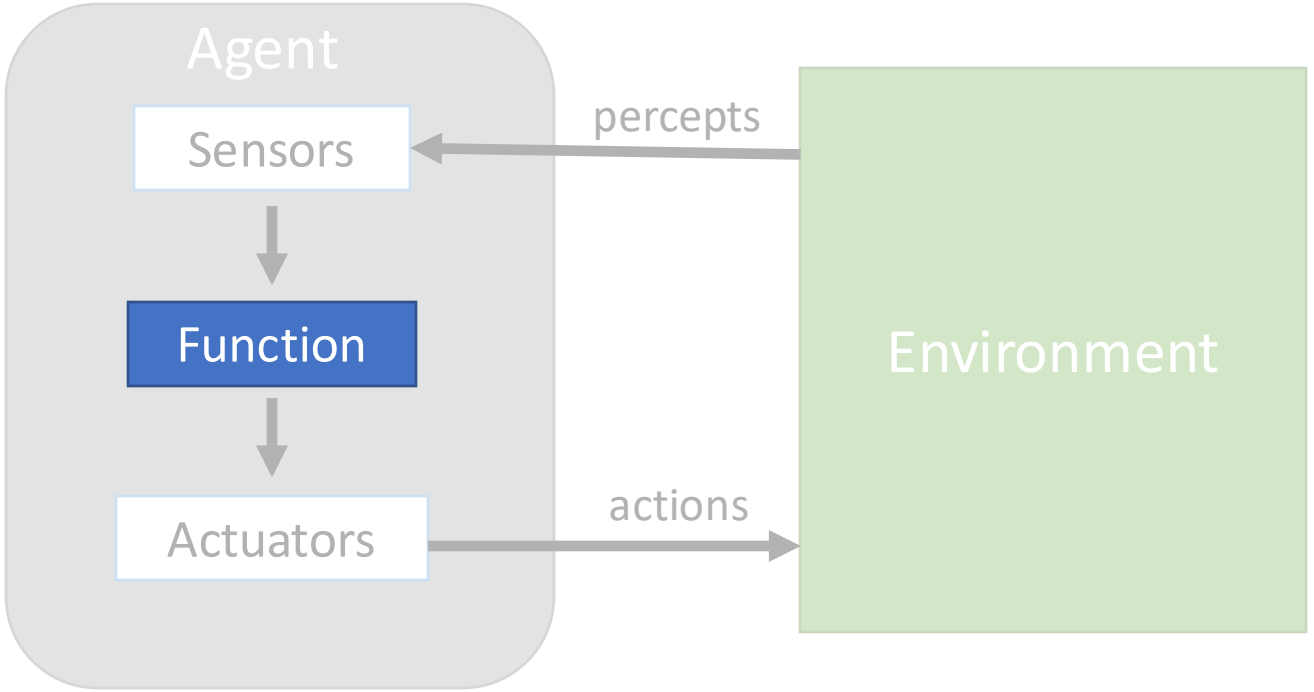
Properties of Task Environment

In this course, we will not deal with environments that are:

- Partially-observable
- Stochastic
- Dynamic

Will be taught in more advanced classes, e.g., CS4246

Intelligent Agents



The agent function maps from percept histories $\mathcal{P}$ to actions $\mathcal{A}$:

$$f: \mathcal{P} \rightarrow \mathcal{A}$$

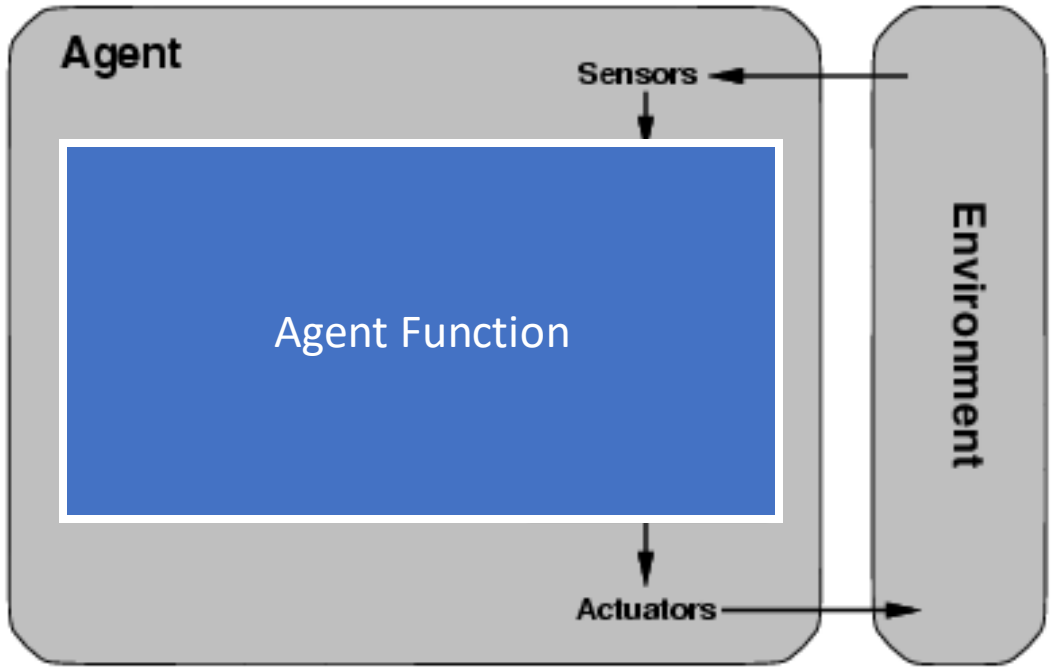
The agent program is an implementation on the physical architecture to produce function f

The Structure of Agents

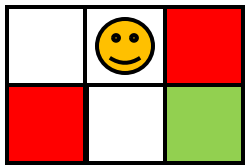
An agent is completely specified by the **agent function**.

Common agent structures

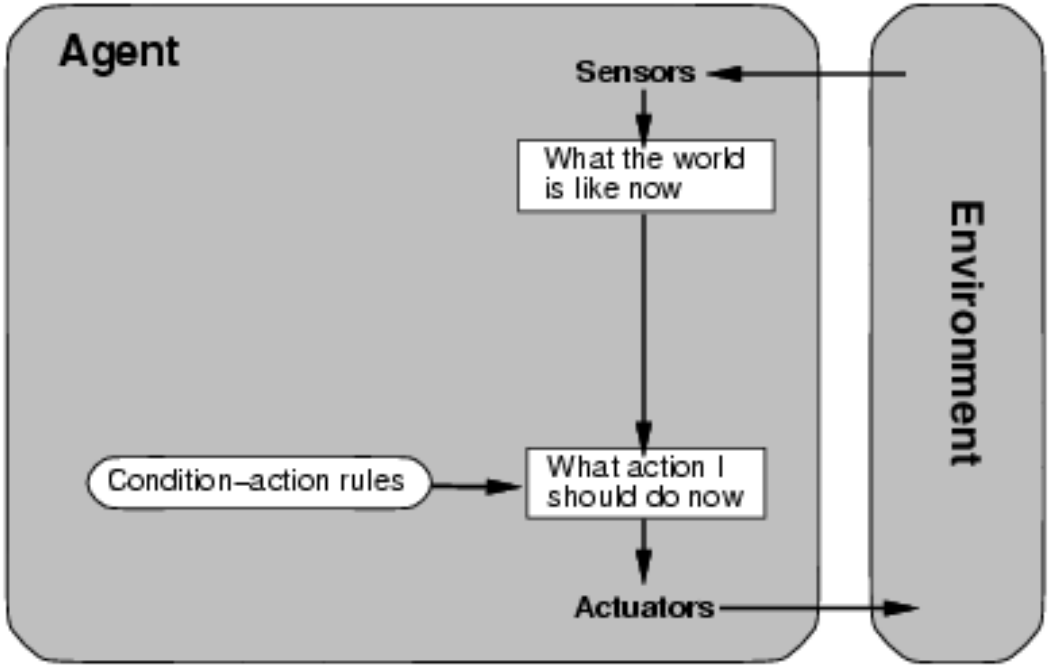
- Simple reflex agents
- Goal-based agents
- Utility-based agents
- Learning agents



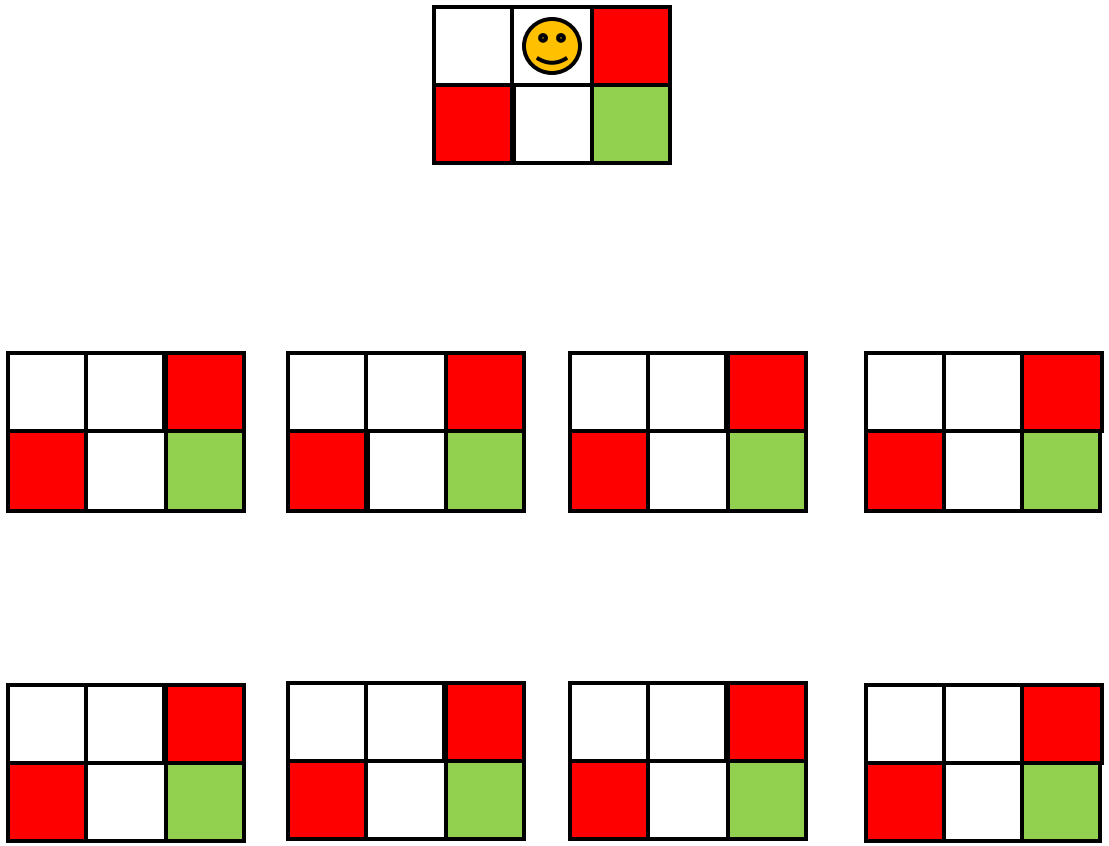
Simple Reflex Agent

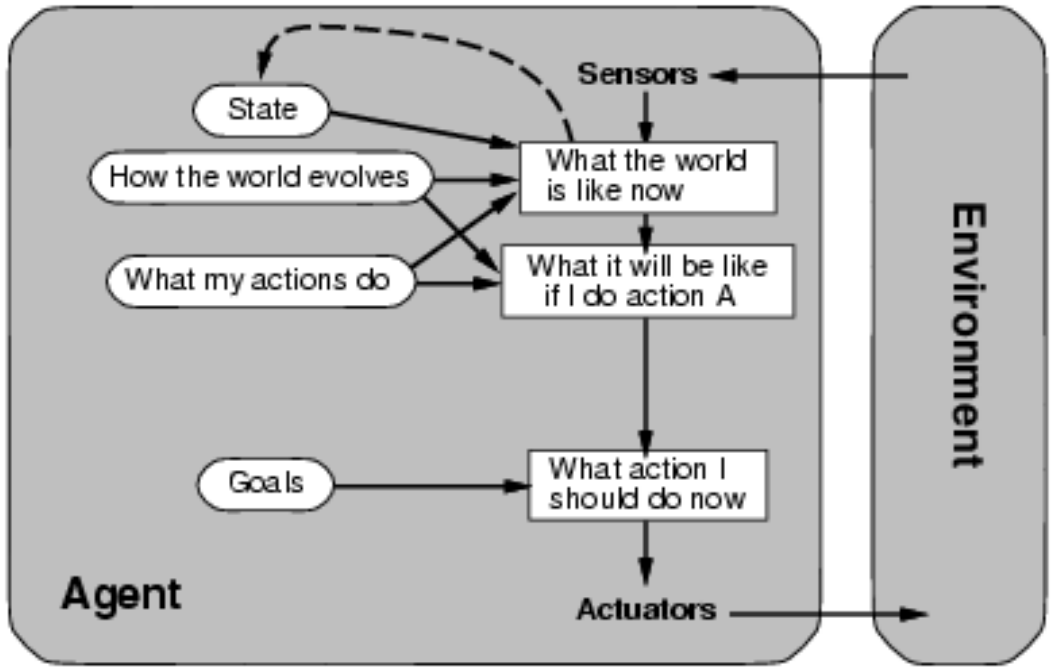


Condition-action rule
If up empty: up
If right empty: right
If down empty: down
If left empty: left

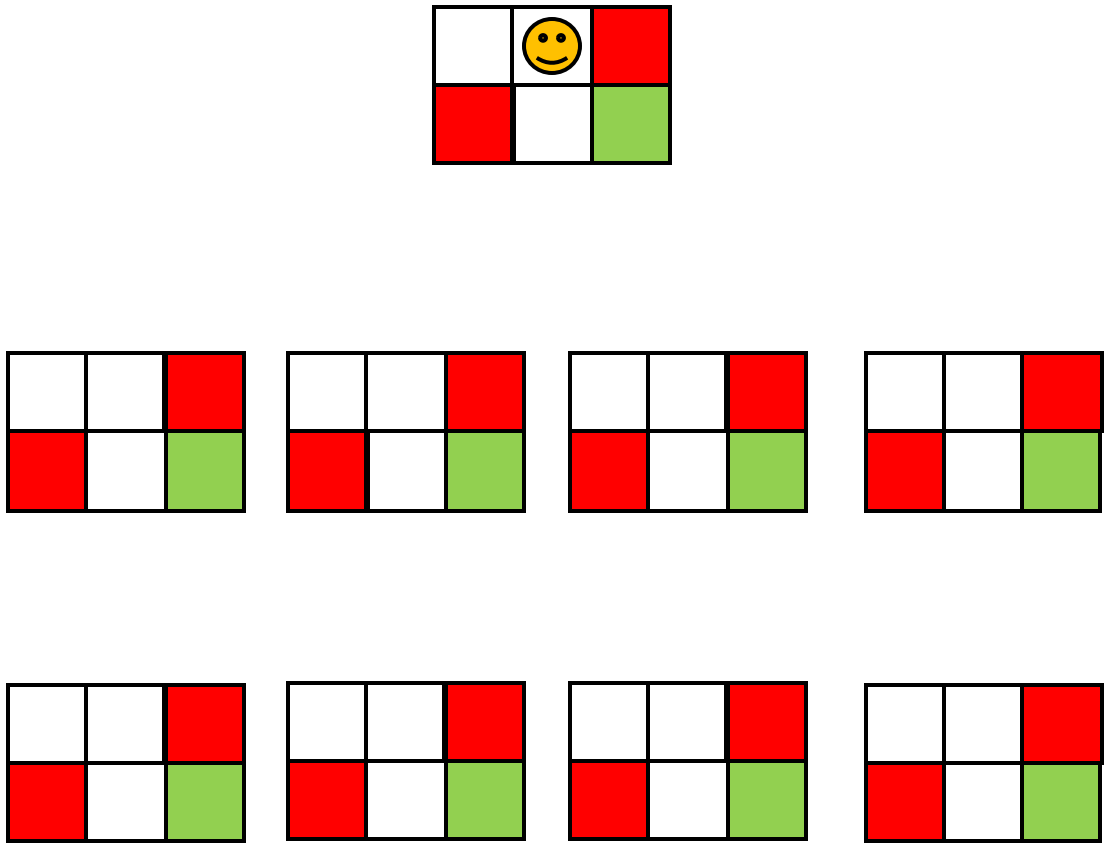


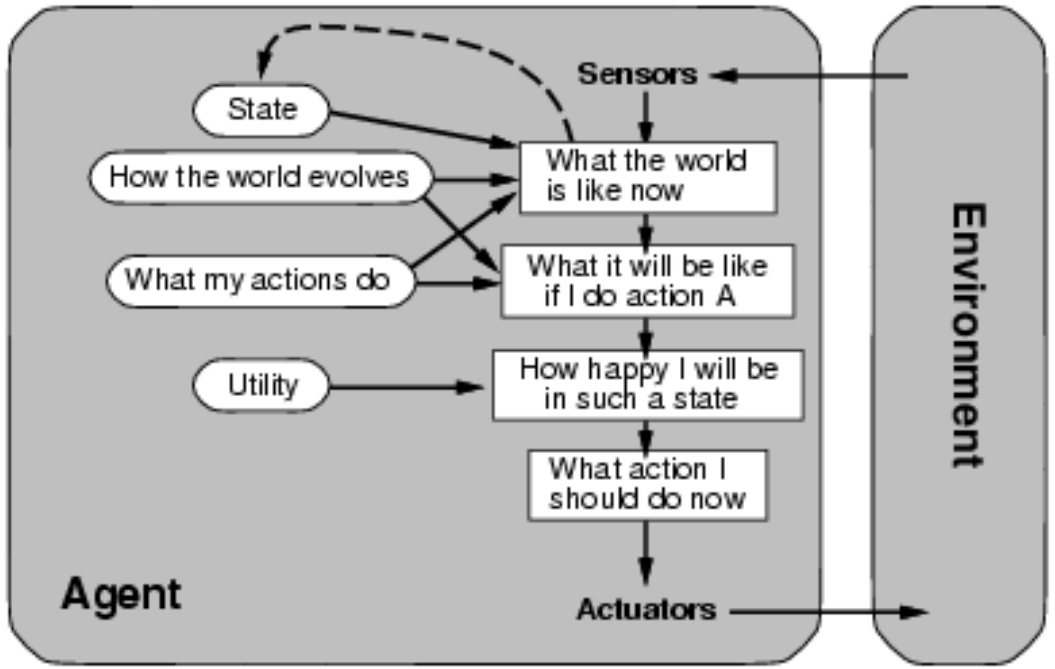
Goal-based Agent



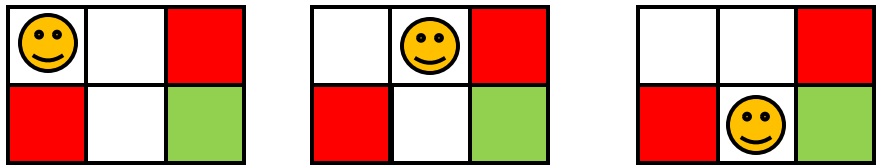


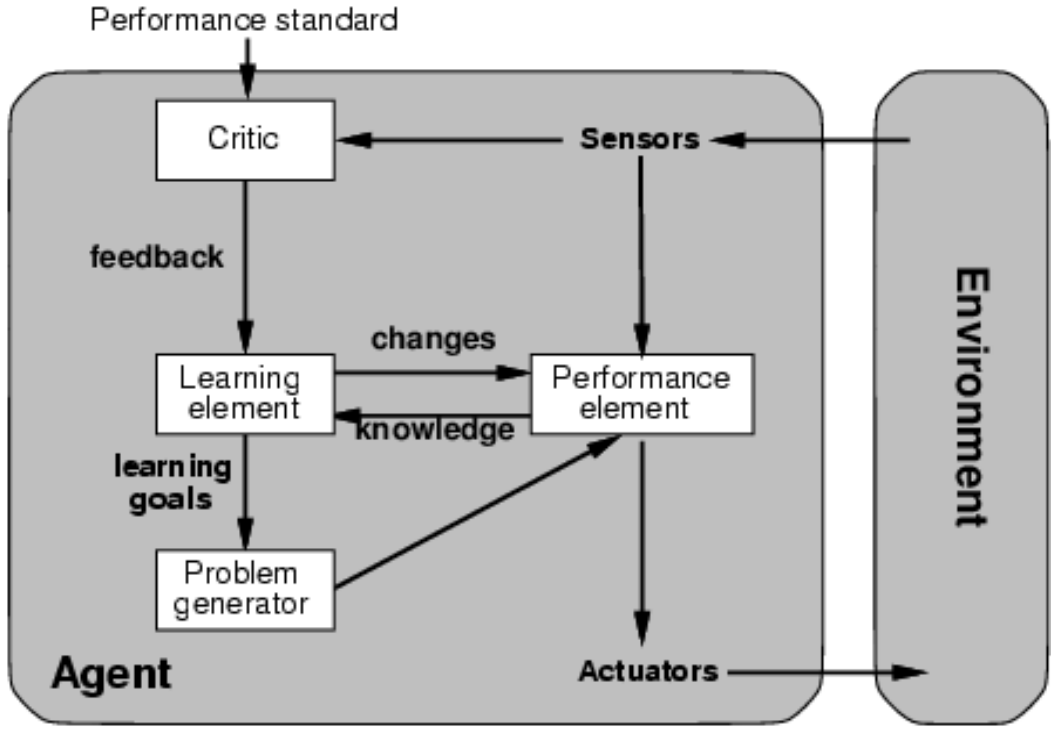
Utility-based Agent





Learning Agent





Other Agent Structures

- Model-based reflex agent
- Combinations of common agent structures (e.g., Alpha Go)

Intelligent Agents: Exploration vs Exploitation

Learn more about the world

Maximize gain based on current knowledge

Credit: Axios

Intelligent Agents: Exploration vs Exploitation

Learn more about the world

Maximize gain based on current knowledge

VS

Credit: Singapore Magazine

Credit: sethlui.com

Summary

- AI: computers trying to **behave like humans**
- **PEAS** Framework:
  - **P**erformance measure: define “goodness” of a solution
  - **E**nvironment: define what the agent can and cannot do
  - **A**ctuators: outputs
  - **S**ensors: inputs
- Agent function is sufficient to define an AI agent
- Common agent structures:
  - Reflex, goal-based, utility-based, learning
  - There are others! Can mix-and-match!
- Exploration vs exploitation

Coming Up Next Week

- **Formulating search problems**
- **Uninformed search algorithms**
  - Breadth-first search
  - Depth-first search
  - Uniform-cost search
  - …
- **Informed search algorithms**
  - Greedy best-first search
  - A* Search
  - …

To Do

- **Lecture Training 1**
  - +250 EXP
  - +100 Early bird bonus
- **Problem Set 0 (completely optional!)**
  - +700 bonus EXP, until 8 February (end of week 4)

Appendix

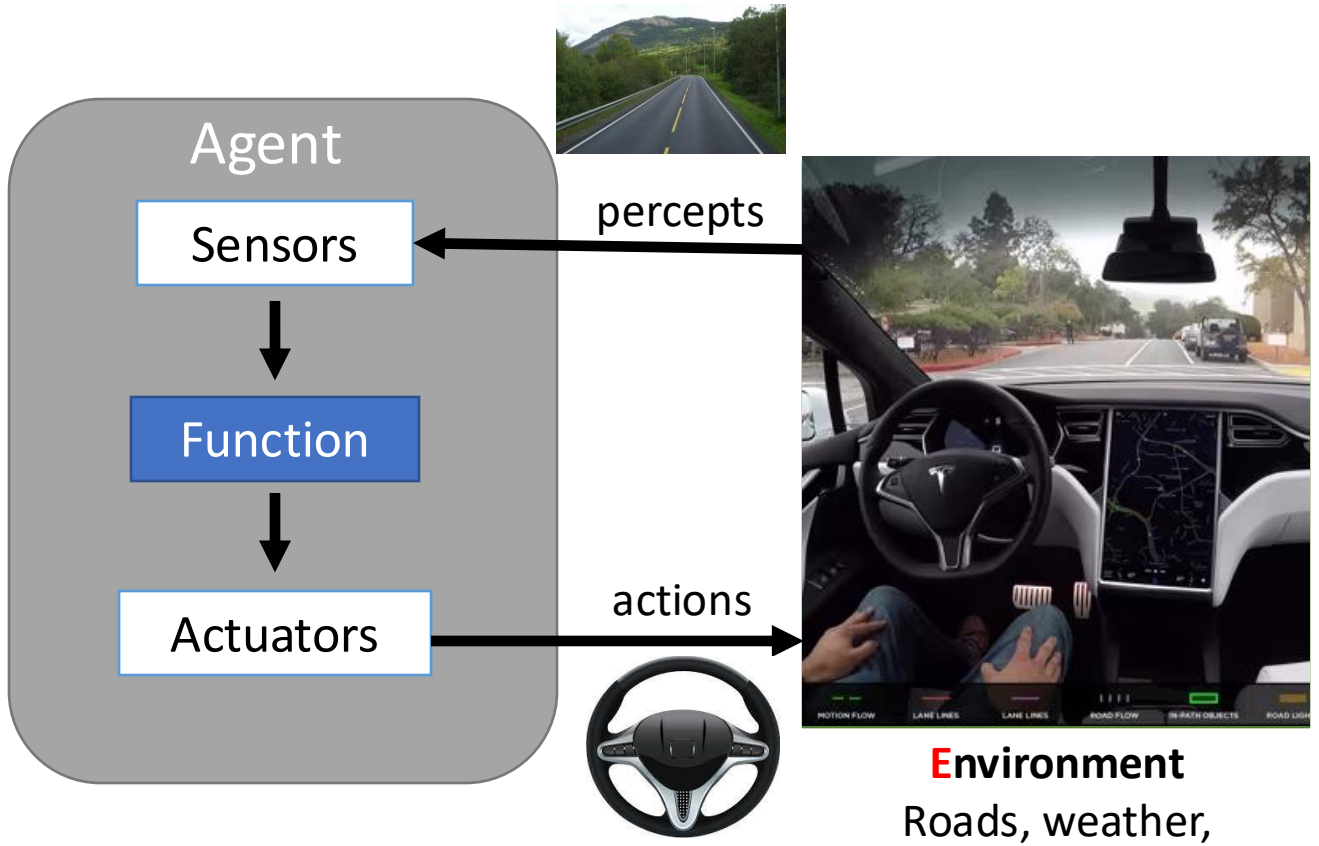
Intelligent Agents: Autonomous Driving

Sensors

- Camera
- LIDAR
- Speedometer
- …

Actuators

- Steering wheel
- Accelerator
- Brake
- …



Environment
Roads, weather,
vehicles, obstacles

Performance Measure

- Safety
- Speed
- Legal
- Comfort
- …

PEAS

Performance Measure, **E**nvironment, **A**ctuators, **S**ensors

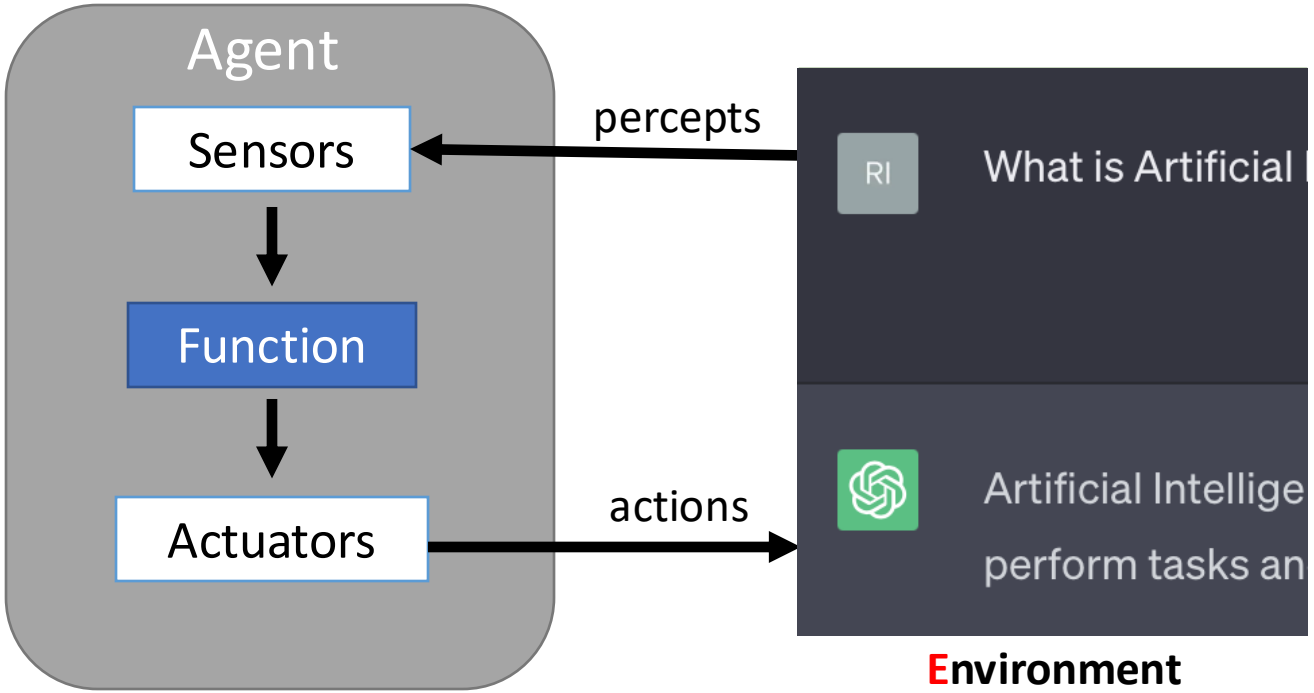
Intelligent Agents: AI Assistant/Chatbot

Sensors

- Text input
- Chat history
- Context
- …

Actuators

- Text output
- Image output
- API
- …



Environment
Chat interface, third-party plugins, user

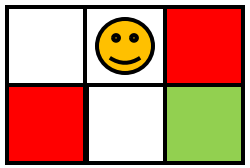
Performance Measure

- Correctness
- Conciseness
- Legal
- Safety
- …

PEAS

Performance Measure, **E**nvironment, **A**ctuators, **S**ensors

Simple Reflex Agent



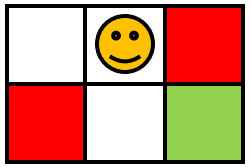
Condition-action rule

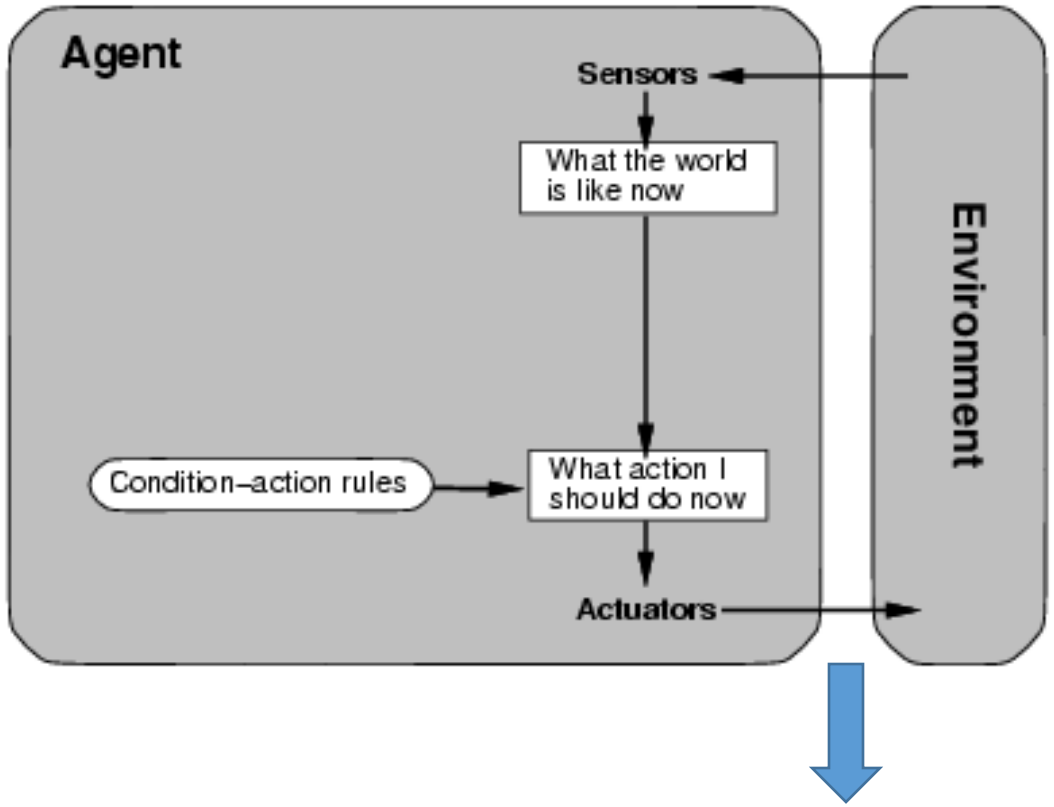
If up empty: up

If right empty: right

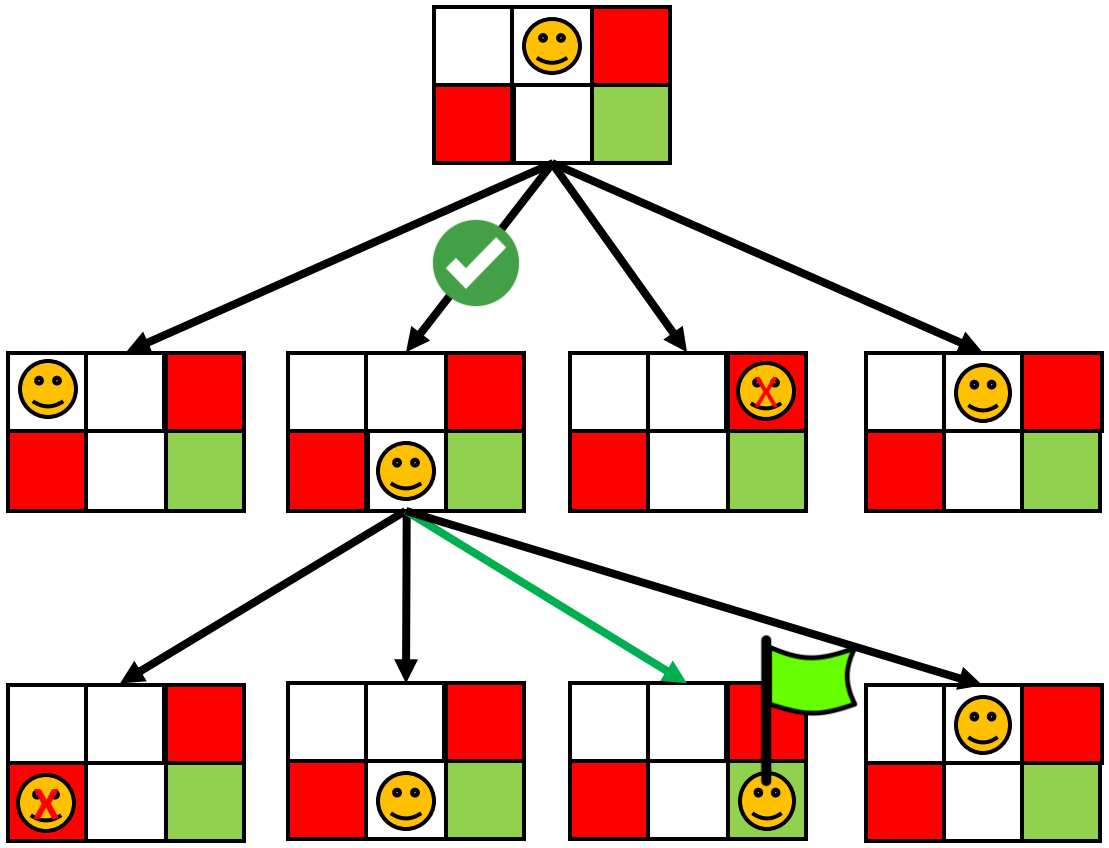
If down empty: down

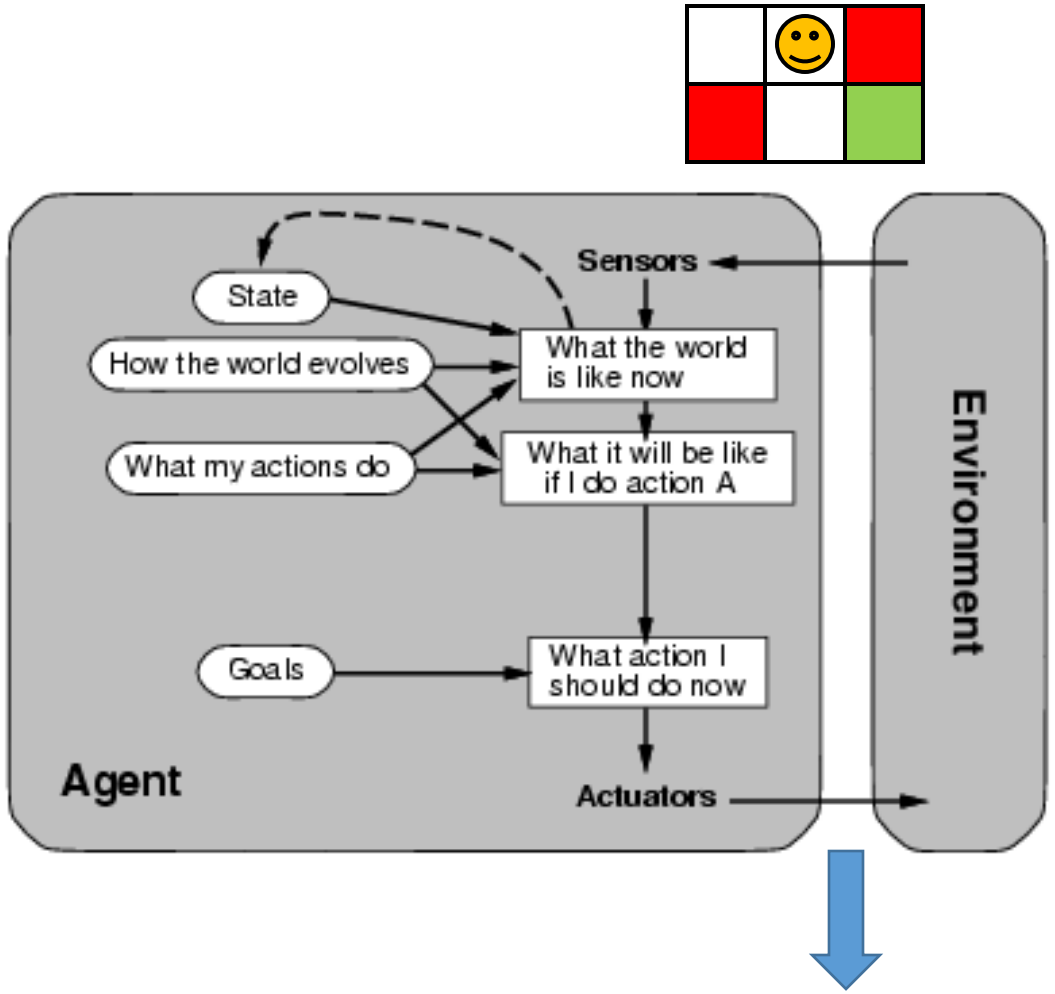
If left empty: left



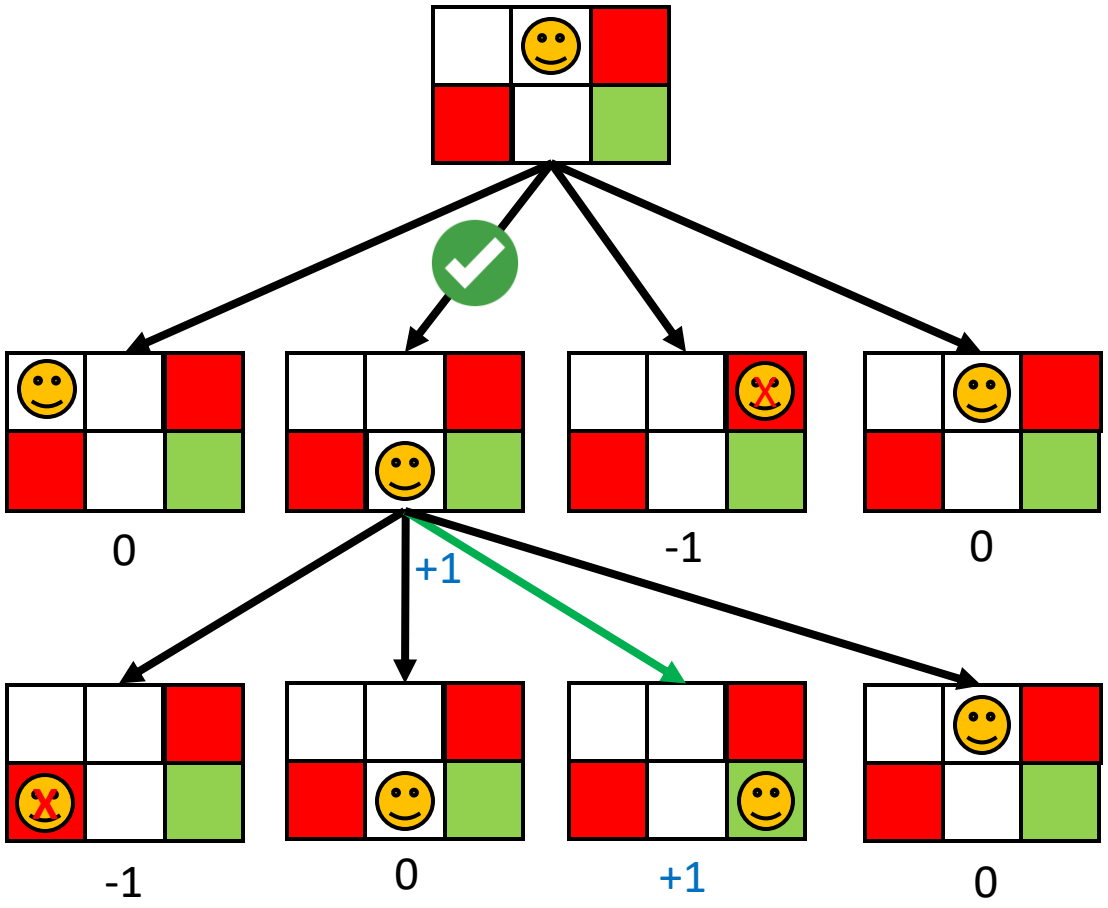


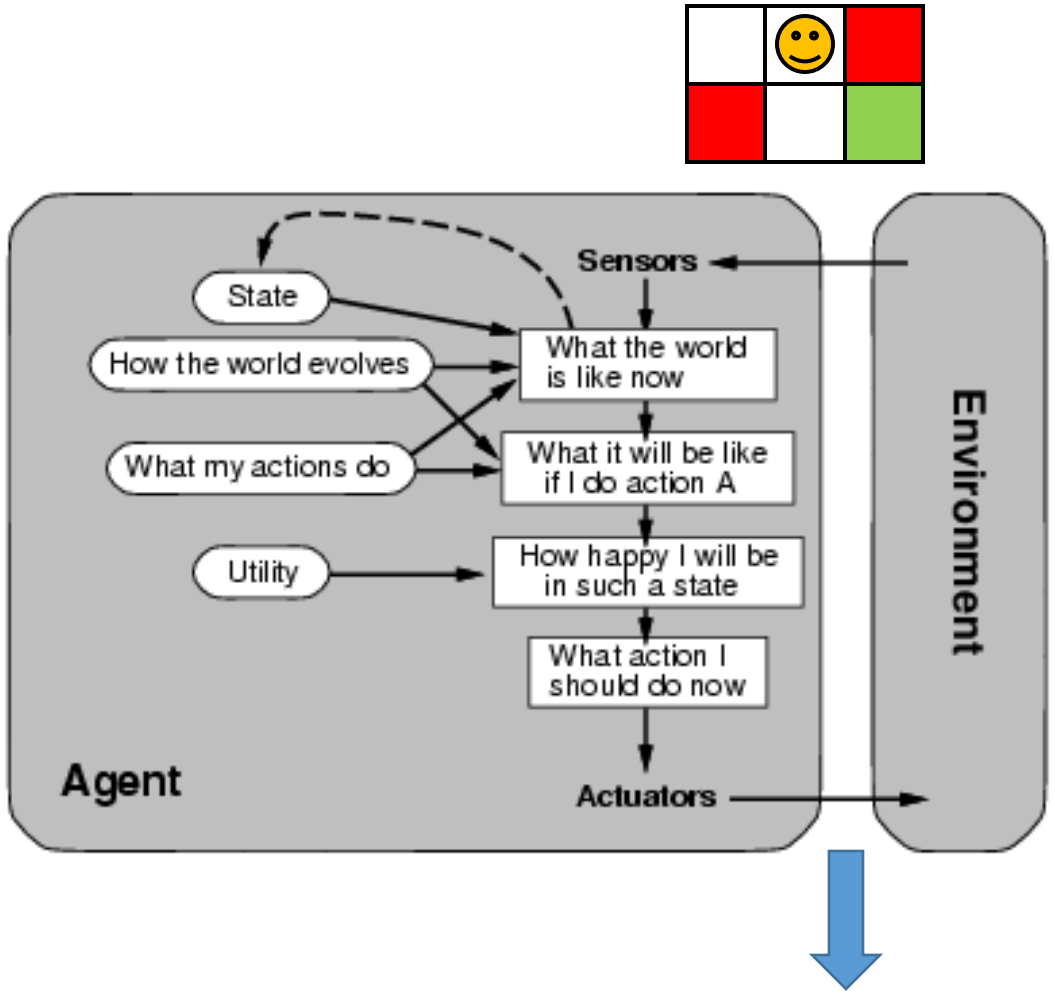
Goal-based Agent



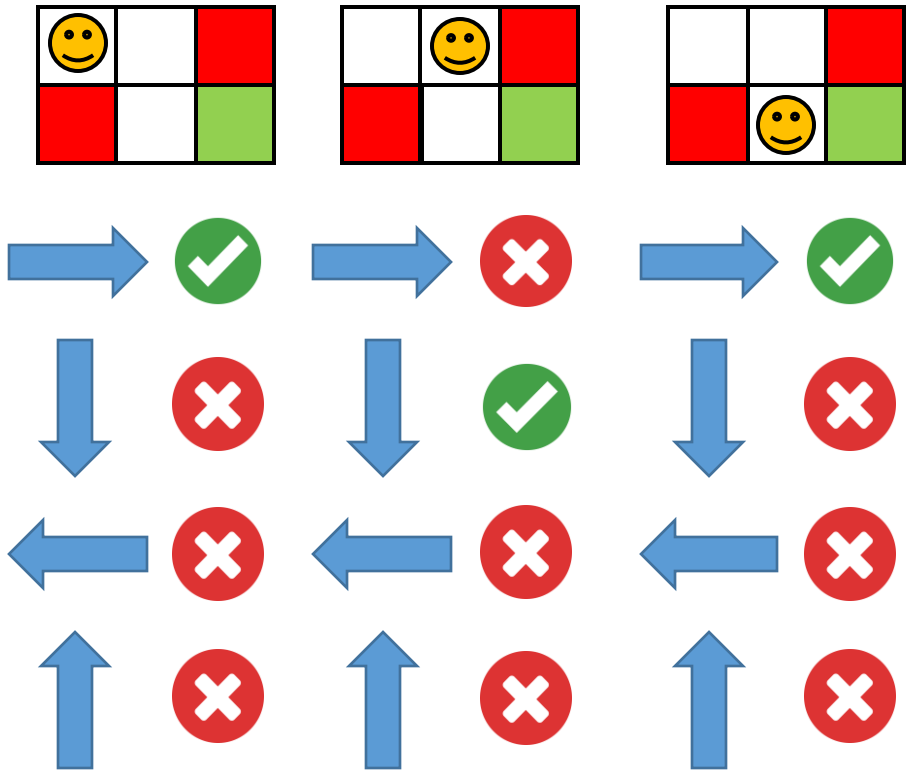


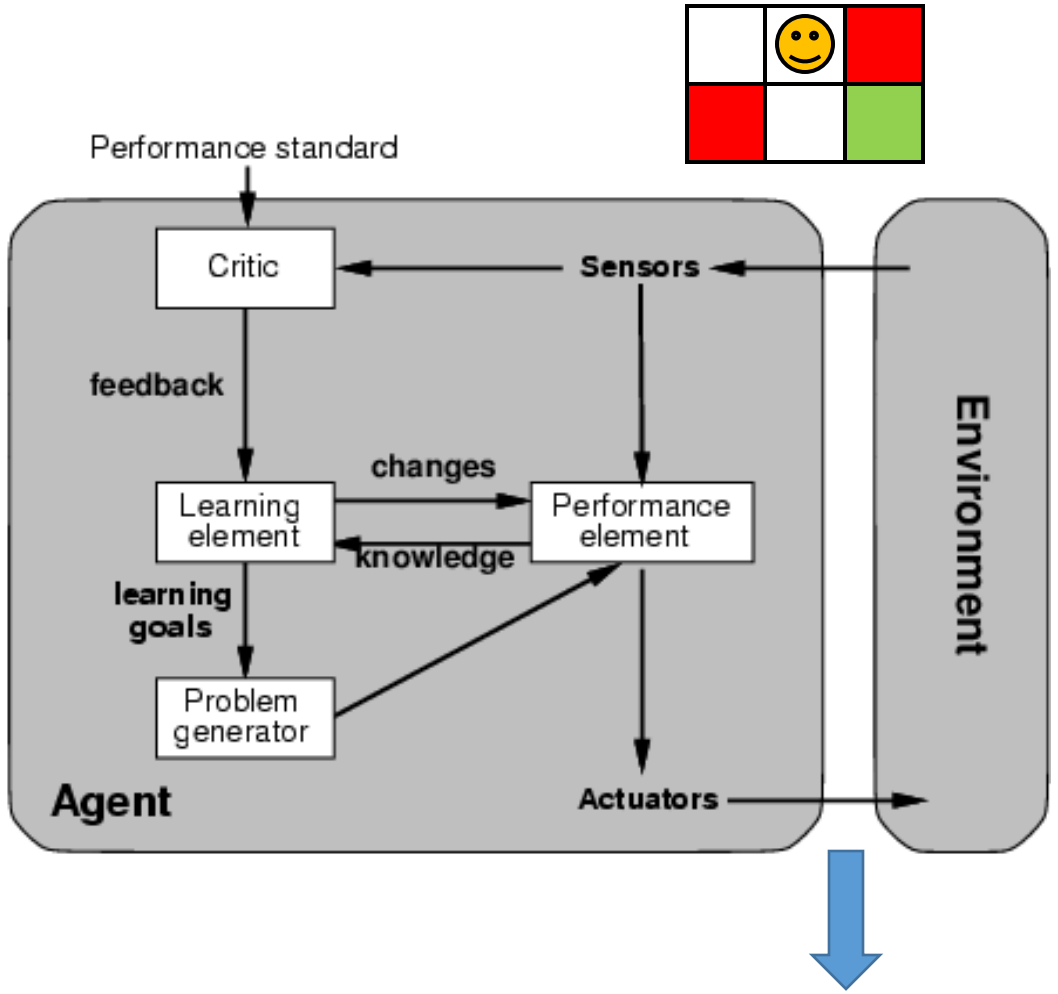
Utility-based Agent





Learning Agent





Can be reflex, model-based, goal-based, and utility-based