Reinforcement learning

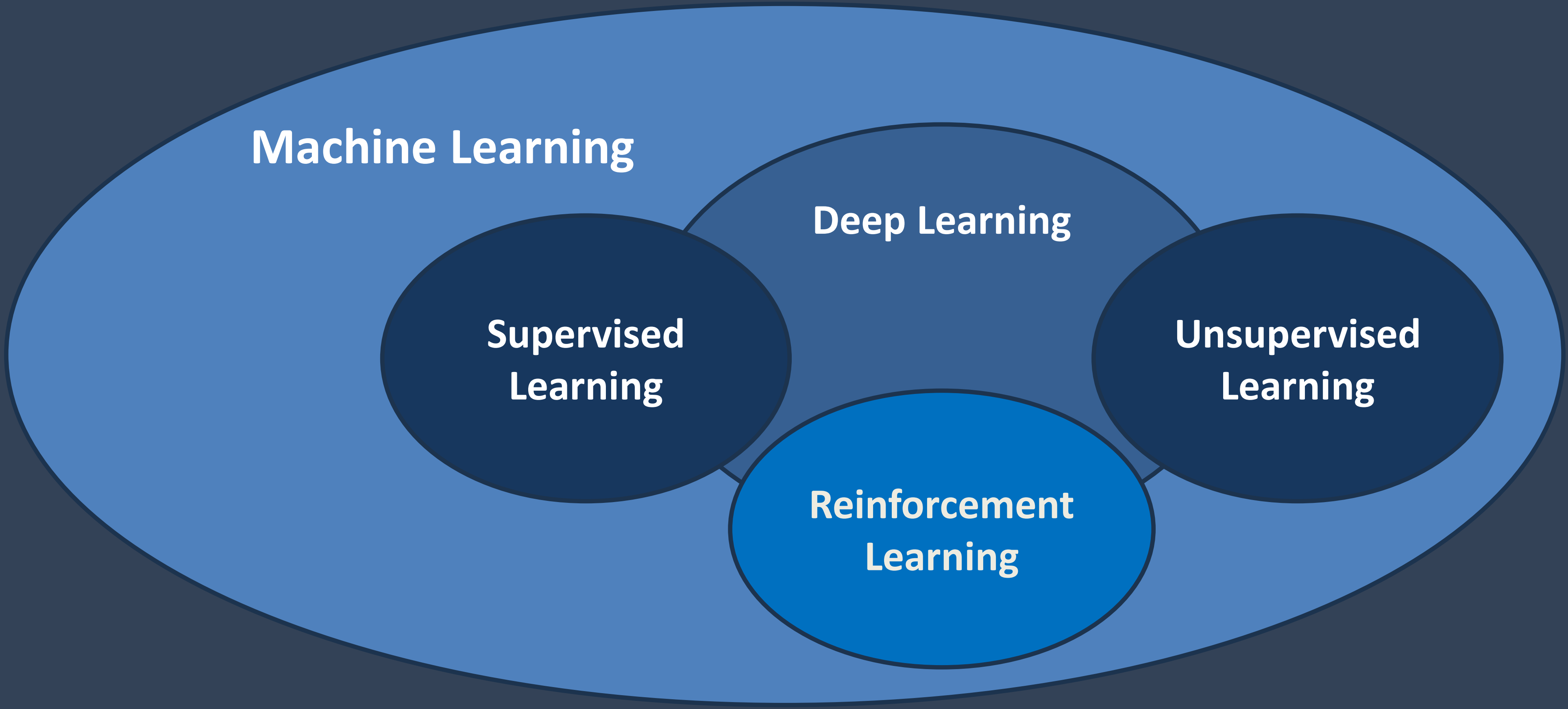
Professor: Li-Der Chou
Presenter: Chih-Jou Tai

Outline

- Introduction
- Methodology
- Applications
- Pros and Cons
- References

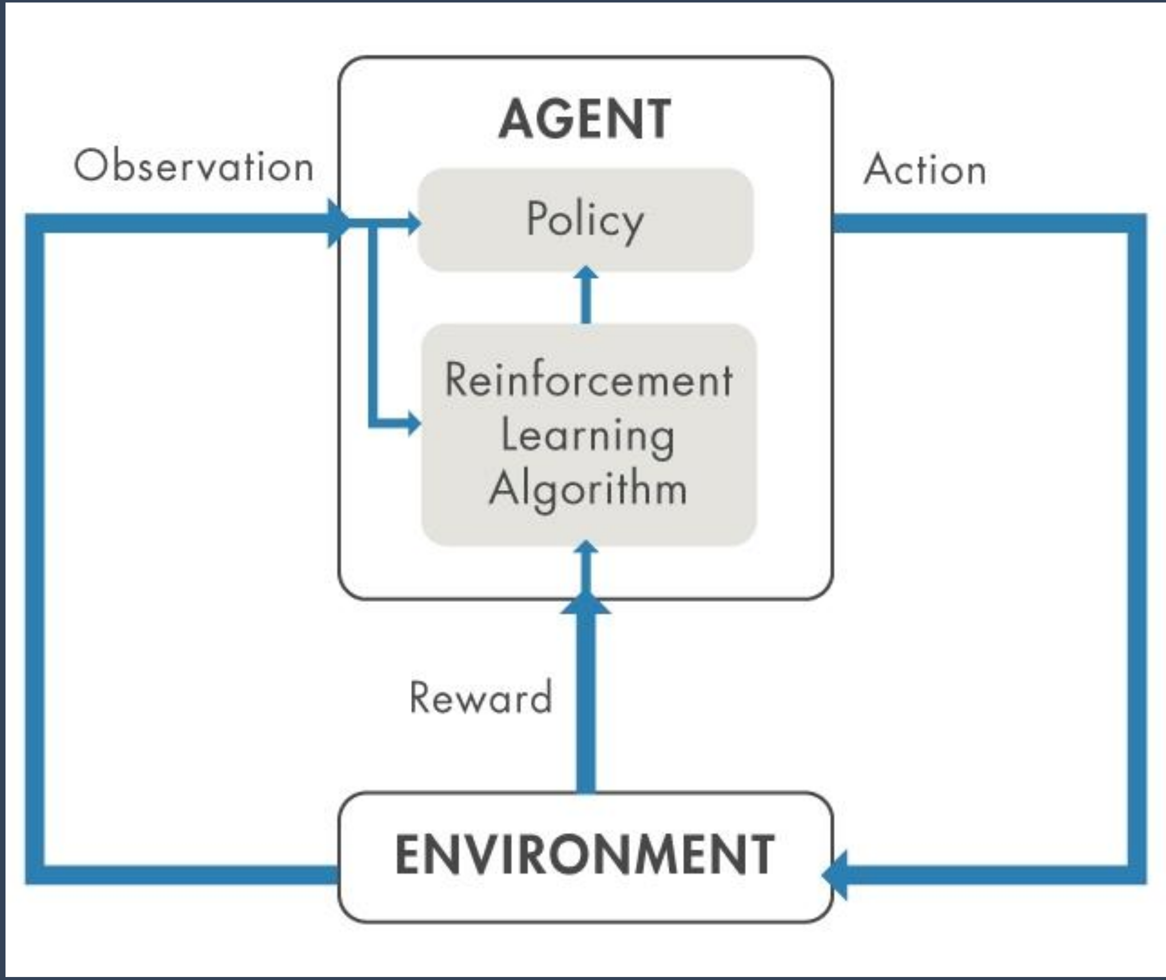
Introduction

- Reinforcement learning is a subfield within machine learning



Introduction

- It involves the computer learning to perform a task correctly through continuous interaction with a dynamic environment, using trial and error.



Outline

- Introduction
- Methodology
- Applications
- Pros and Cons
- References

Methodology

1. Environment Setup
2. Reward Definition
3. Agent Creation
4. Agent Training and Validation
5. Policy Deployment

Methodology-Environment Setup

- Define an environment in which the agent can learn, including specifying the interface between the agent and the environment.

- Simulation Environment: A virtual environment simulated by a computer, used for safe, controllable, and cost-effective experimentation and testing.

- Experimental Environment: An experimental environment is a physical environment in the real world, providing a more realistic context, but with lower control and safety and higher costs.

Methodology-Reward Definition

- Define the agent's reward signals and explain how they are calculated, including factors like performance metrics, costs, risks, and more.
- Discounting: Discounting is about valuing immediate rewards more than delayed rewards in reinforcement learning by using a discount factor (often denoted as γ) between 0 and 1.
- Sparsity: Sparsity in reinforcement learning means that rewards are rare or scarce, making the agent explore and figure out how to obtain them effectively.

Methodology-Agent Creation

- Selecting a policy representation method and choosing an appropriate training algorithm.

- Policy Representation: This step entails deciding how to represent the agent's policy. Common methods include using neural networks or lookup tables.

- Training Algorithm: Modern reinforcement learning often uses neural networks or Q-Learning for complex tasks with large state and action spaces. It determines how the agent adjusts its policy during learning.

Methodology-Agent Training and Validation

- Configure training options and train the agent to adapt its policy. Validation of the trained policy is typically performed through simulations.
- Training Configuration: Including parameters like learning rates and exploration strategies.
- Policy Validation: After training, it's essential to assess the agent's policy to ensure it aligns with your goals. Validation is typically performed through simulations or real-world experiments.

Methodology-Policy Deployment

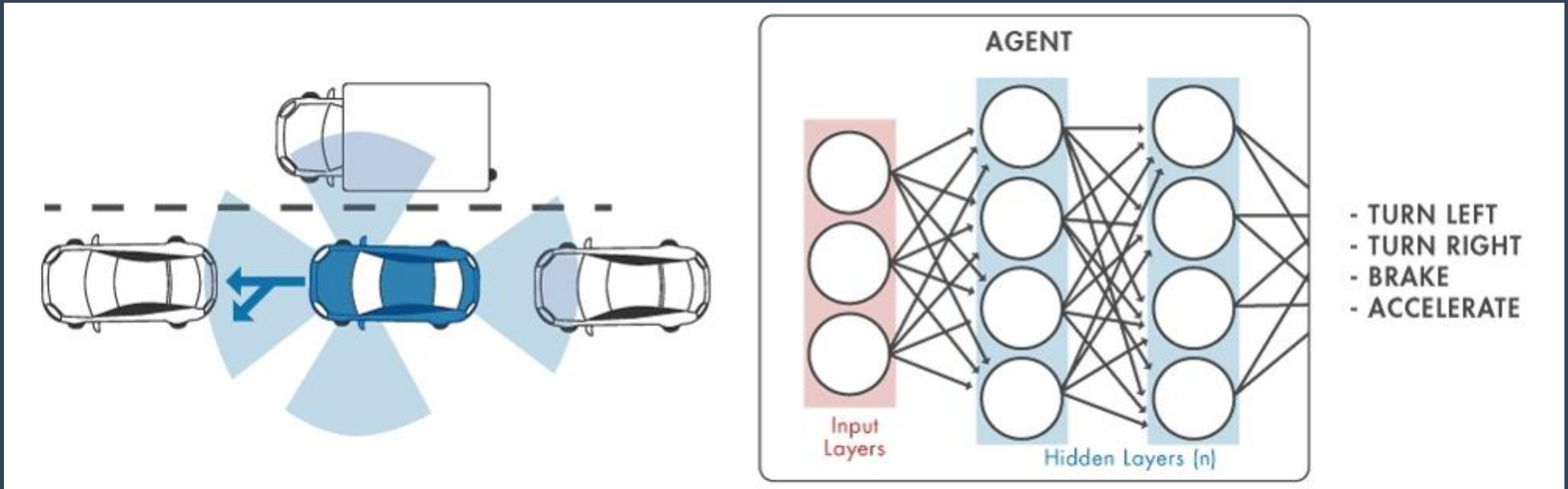
- Determine how the trained policy will be deployed.
- Programming Languages: Common choices include C/C++, CUDA, Python, or specialized languages like TensorFlow or PyTorch.
- Deployment Environment: It can be embedded in a robot, integrated into software, or employed in a simulation.
- Testing and Optimization: Fine-tune and optimize it as needed to ensure it performs effectively in real-world scenarios.
- Monitoring and Maintenance: Monitor the policy's performance and be prepared to make adjustments or updates as necessary.

Outline

- Introduction
- Methodology
- Applications
- Pros and Cons
- References

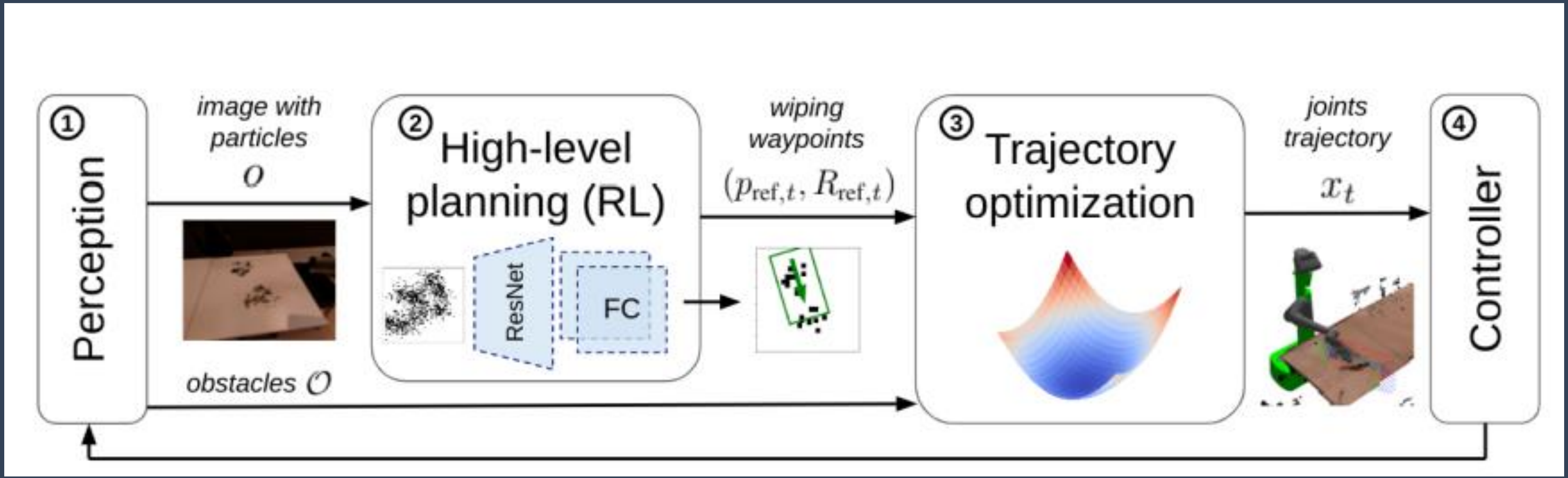
Applications

- **Autonomous Driving**
  - In autonomous driving, the agent learns to navigate dynamic environments using sensors, policies driven by neural networks, and iterative reinforcement learning with rewards.



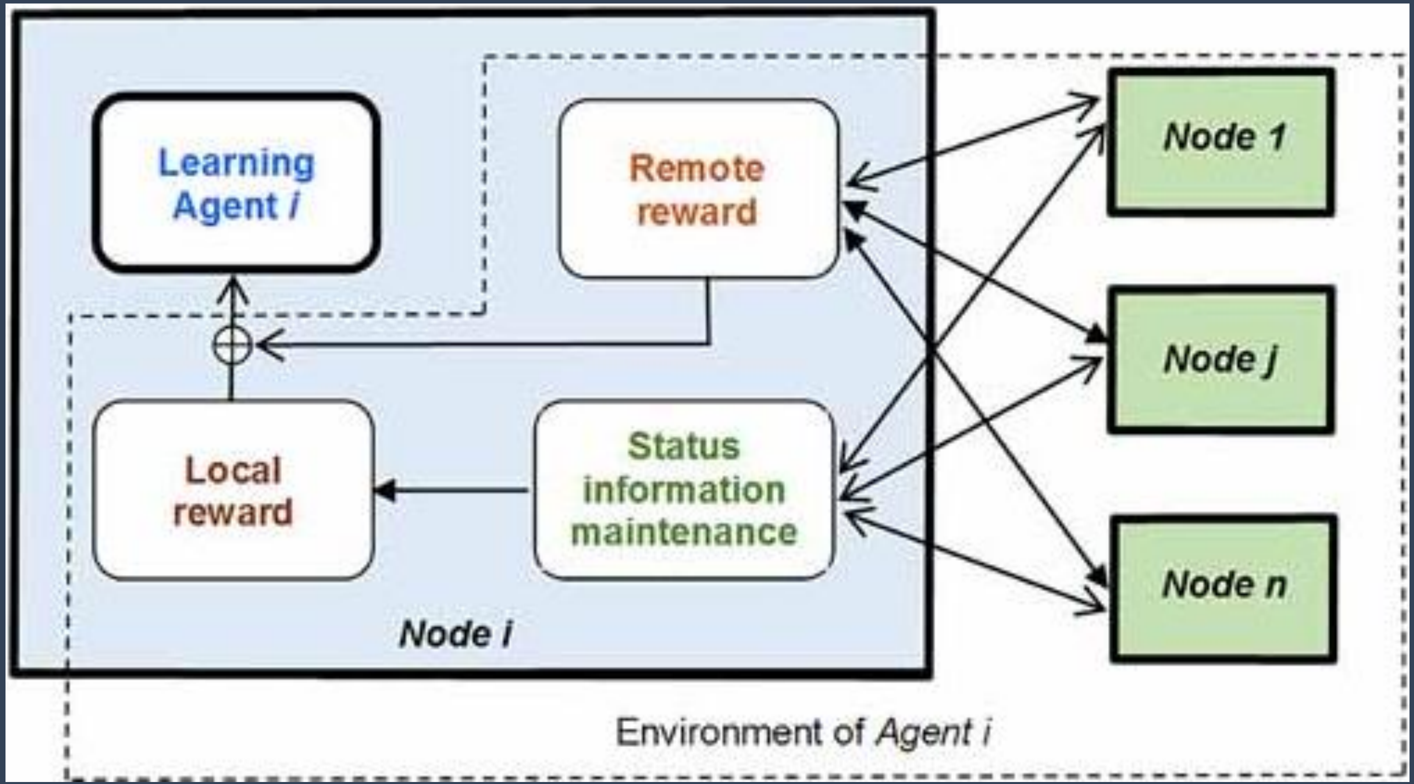
Applications

- **Robotic Table Wiping**
  - This robot autonomously cleans tables by wiping away spills and crumbs using a combination of vision-based reinforcement learning and trajectory optimization for real-world deployment.



Applications

- **Network Packet Routing**
  - Agents (routers or switches) learn how to choose the optimal path to minimize packet transmission time or minimize network congestion.



Outline

- Introduction
- Methodology
- Applications
- Pros and Cons
- References

Pros

- Versatility: Applies to various tasks and domains, offering a versatile problem-solving approach.
- Autonomy: Agents learn and decide independently once the environment and algorithms are set up.
- Adaptability: Agents refine strategies through experience, enhancing performance over time.

Cons

- Sample Inefficiency: Often demands substantial training data, leading to time and resource intensiveness.
- Complex Problem Setup: Designing problems correctly can be challenging, requiring multiple design iterations.
- Black-Box Nature: Trained neural network policies can be hard to interpret, affecting transparency.

Outline

- Introduction
- Methodology
- Applications
- Pros and Cons
- References

References

- 強化學習(Reinforcement Learning)：入門指南
  https://www.terasoft.com.tw/support/tech_articles/reinforcement_learning_a_brief_guide.asp
- 強化學習訓練打掃機器人
  https://arxiv.org/abs/2210.10865
- Reinforcement Learning Based Routing in Networks
  https://ieeexplore.ieee.org/document/8701570

Q & A