Introduction of Reinforcement Learning with Related Applications

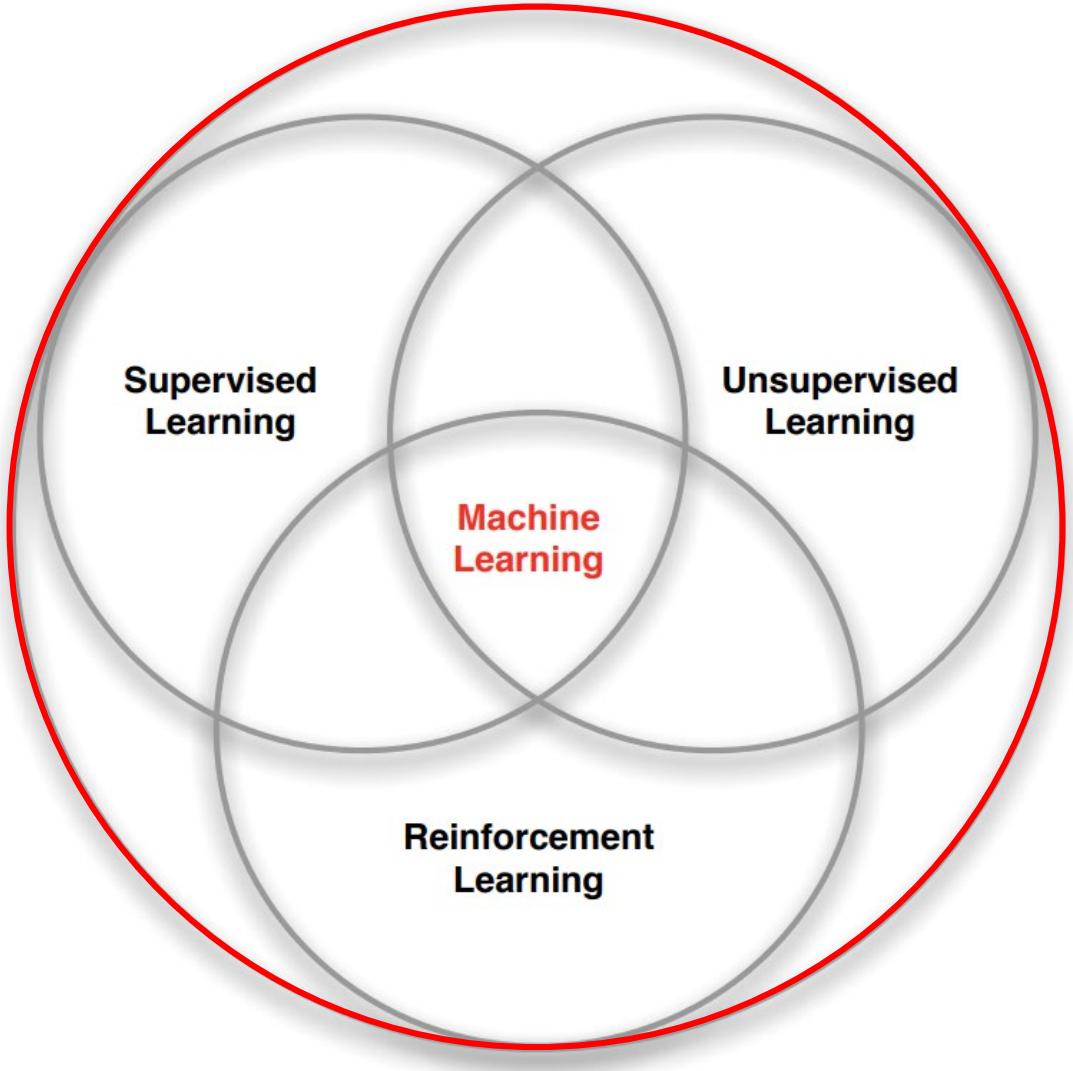
Minsung Yoon
Ph.D. student at SGVR Lab.

KAIST
Korea Advanced Institute of Science and Technology
School of Computing

Contents

- Overview of Reinforcement Learning (RL) Technology
- Applications for Robot Arms
  - Learning-based Initialization of Trajectory Optimization for Path-following Problems of Redundant Manipulators, **ICRA 2023**
  - Towards Safe Remote Manipulation: User Command Adjustment based on Risk Prediction for Dynamic Obstacles, **ICRA 2023**
- Applications for Quadruped Robots
  - Enhancing Navigation Efficiency of Quadruped Robots via Leveraging Personal Transportation Platforms, **ICRA 2025**
  - Learning-based Adaptive Control of Quadruped Robots for Active Stabilization on Moving Platforms, **IROS 2024**

Branches of Machine Learning



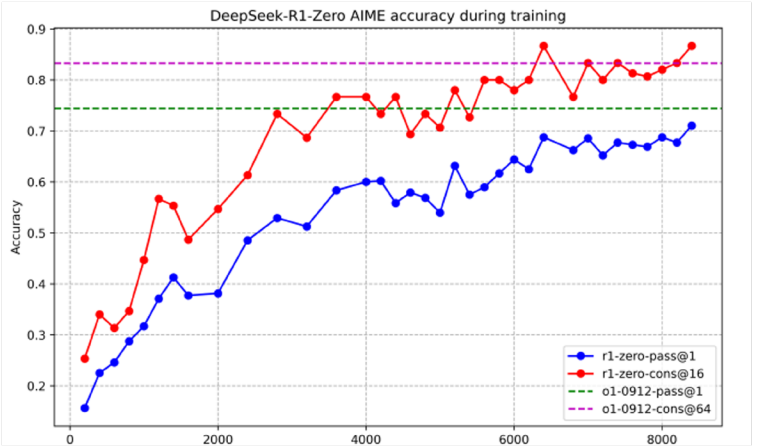
Agents in Reinforcement Learning

Agent?

Robots
(embodied AI)

deepseek

DeepSeek-R1-Zero
: Boost a reasoning capability trough RL

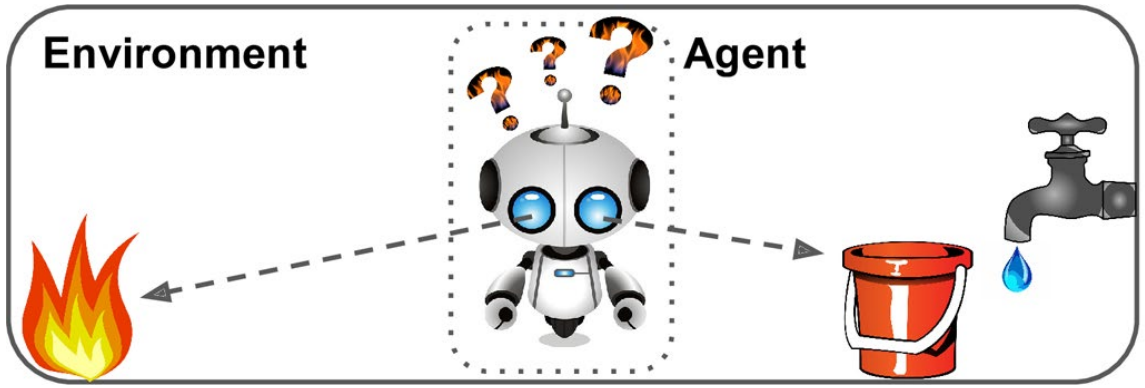


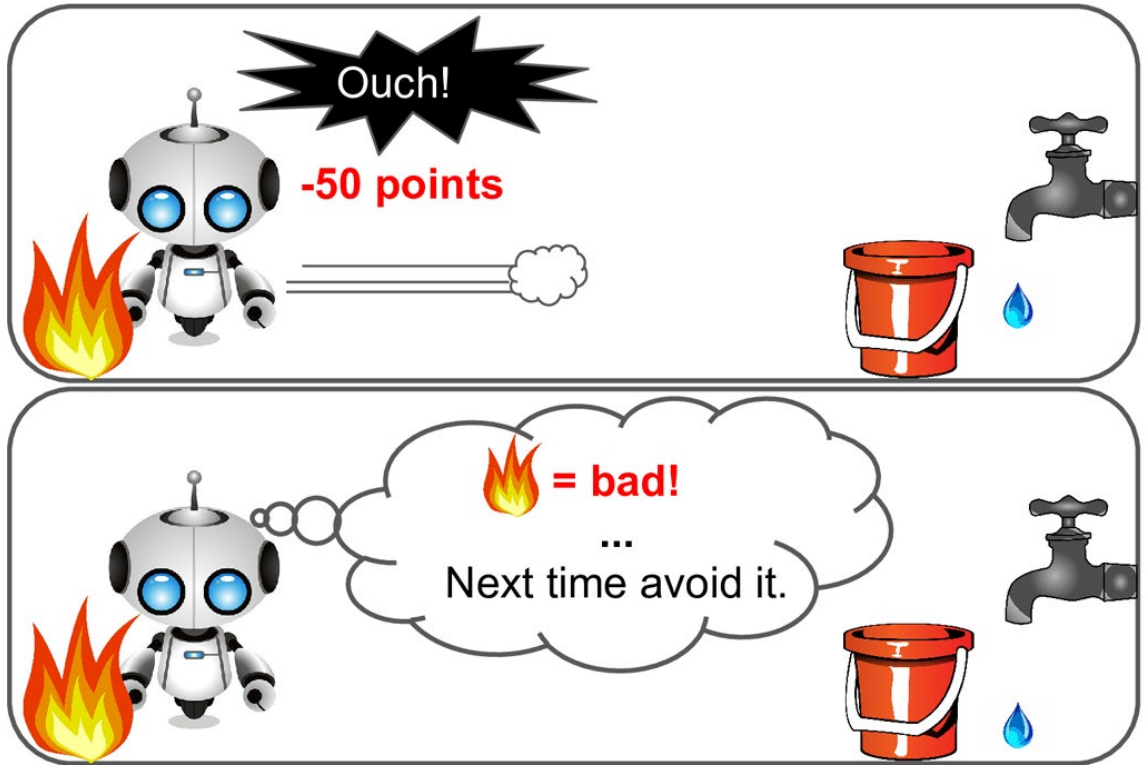
Language model
(software)

Reinforcement Learning

- Reinforcement learning is a type of machine learning where agents learn optimal behaviors through trial and error interactions with an environment.

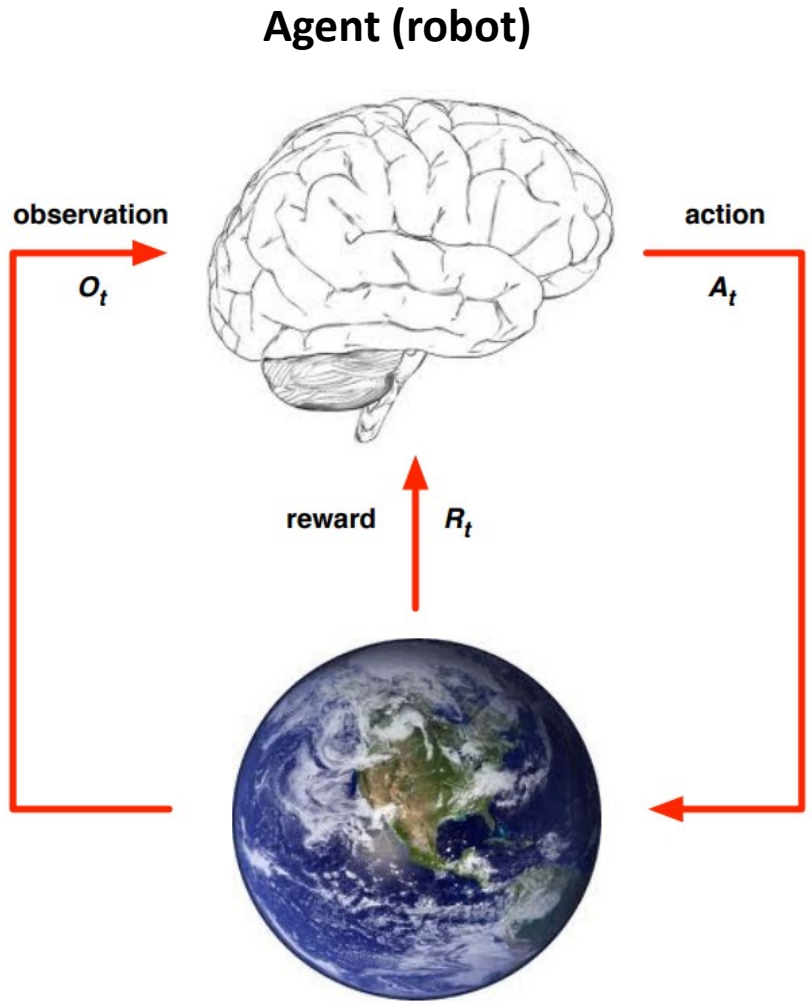
Task: extinguish the fire





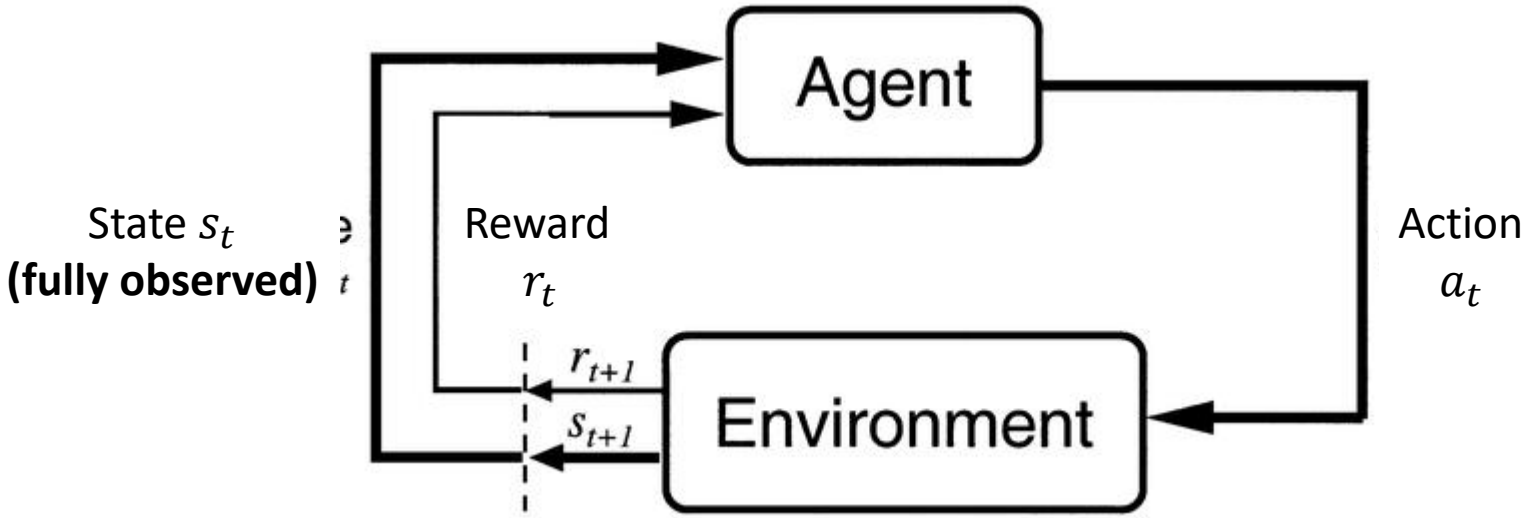
Agent and Environment

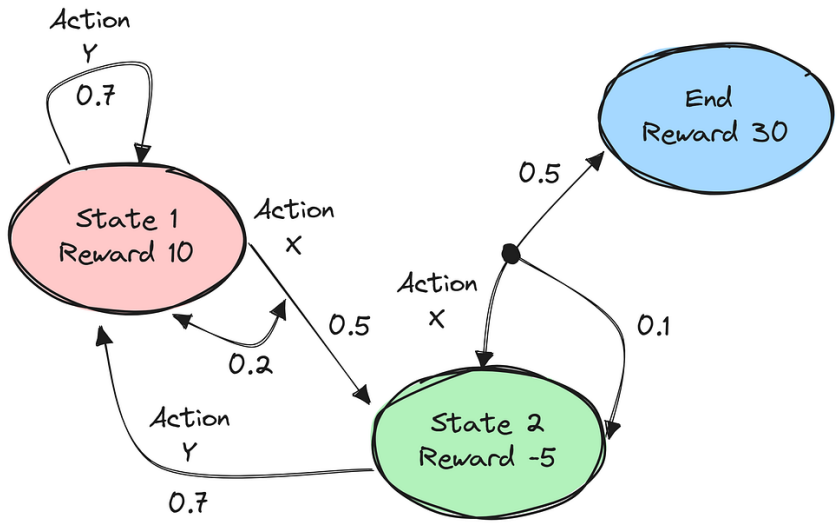
- At each time step t, the agent:
  - Executes action A_t
  - Receives observation O_t (sensing)
  - Receives scalar reward R_t
- The environment:
  - Receives action A_t
  - Emits observation O_{t+1}
  - Emits scalar reward R_{t+1}



Markov Decision Process (MDP)

- MDP is a formal mathematical modeling of RL problem, defined by a 5-tuple.
  - **States (S):** All information describing the agent's situation.
  - **Actions (A):** Available choices the agent can make in each state.
  - **Rewards (R):** Immediate (positive or negative) feedback for an action taken in a state.
  - **Transitions (T):** Probabilities of moving between states after actions.
  - **Discount Factor (γ):** Reduces the value of future rewards compared to immediate ones.



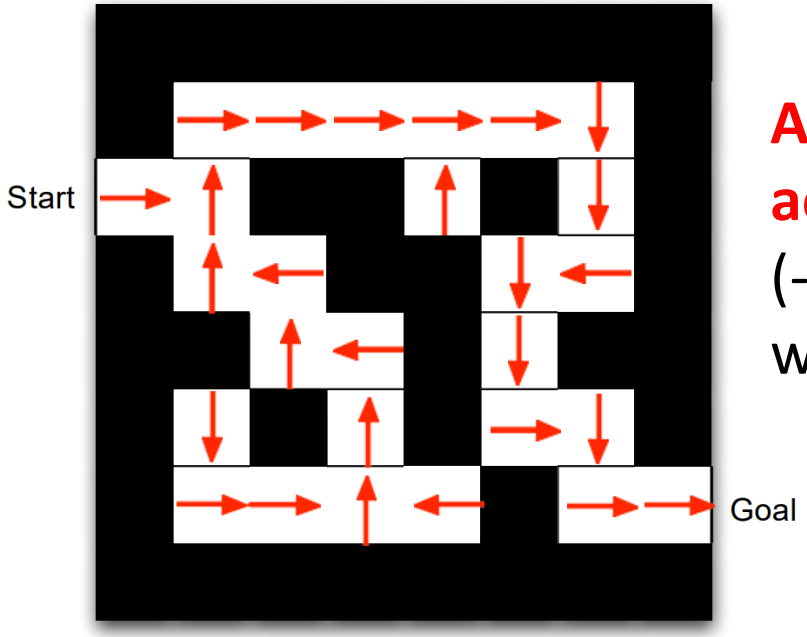


Optimization objective of RL

- The goal of RL is to find an optimal "policy (agent) π^*" which produces the action maximizing the discounted expected future rewards at each state.

$$\pi^* = argmax_\pi J(\pi)$$

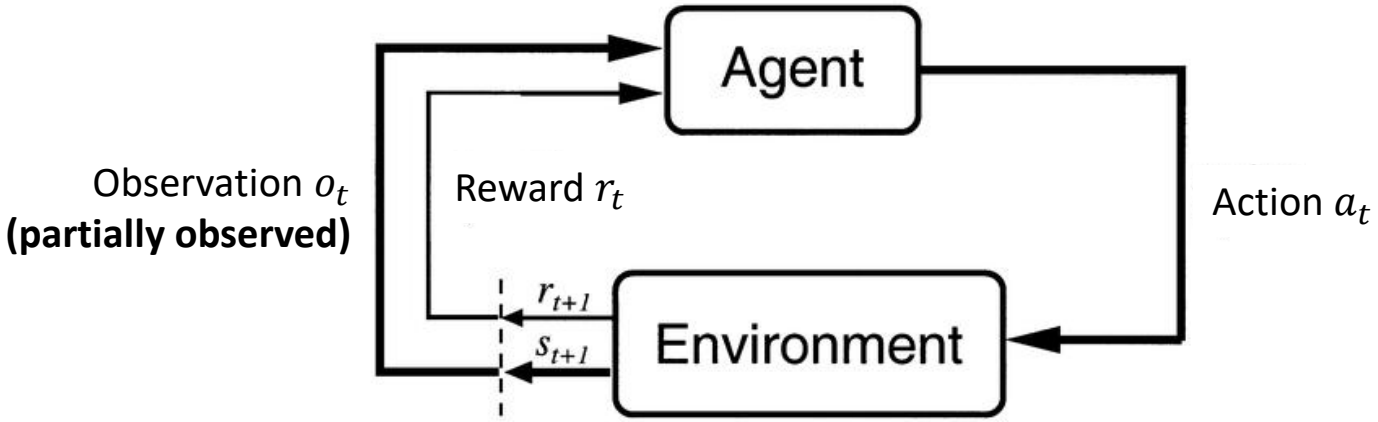
$$J(\pi) = E_{a\sim\pi(s),s'\sim T(s,a)}[\sum_{t=0}^{\infty}\gamma^t R(s,a,s')]$$



Arrows are **induced optimal actions** at each state
(-1 rewards at each step, except when reaching the goal state)

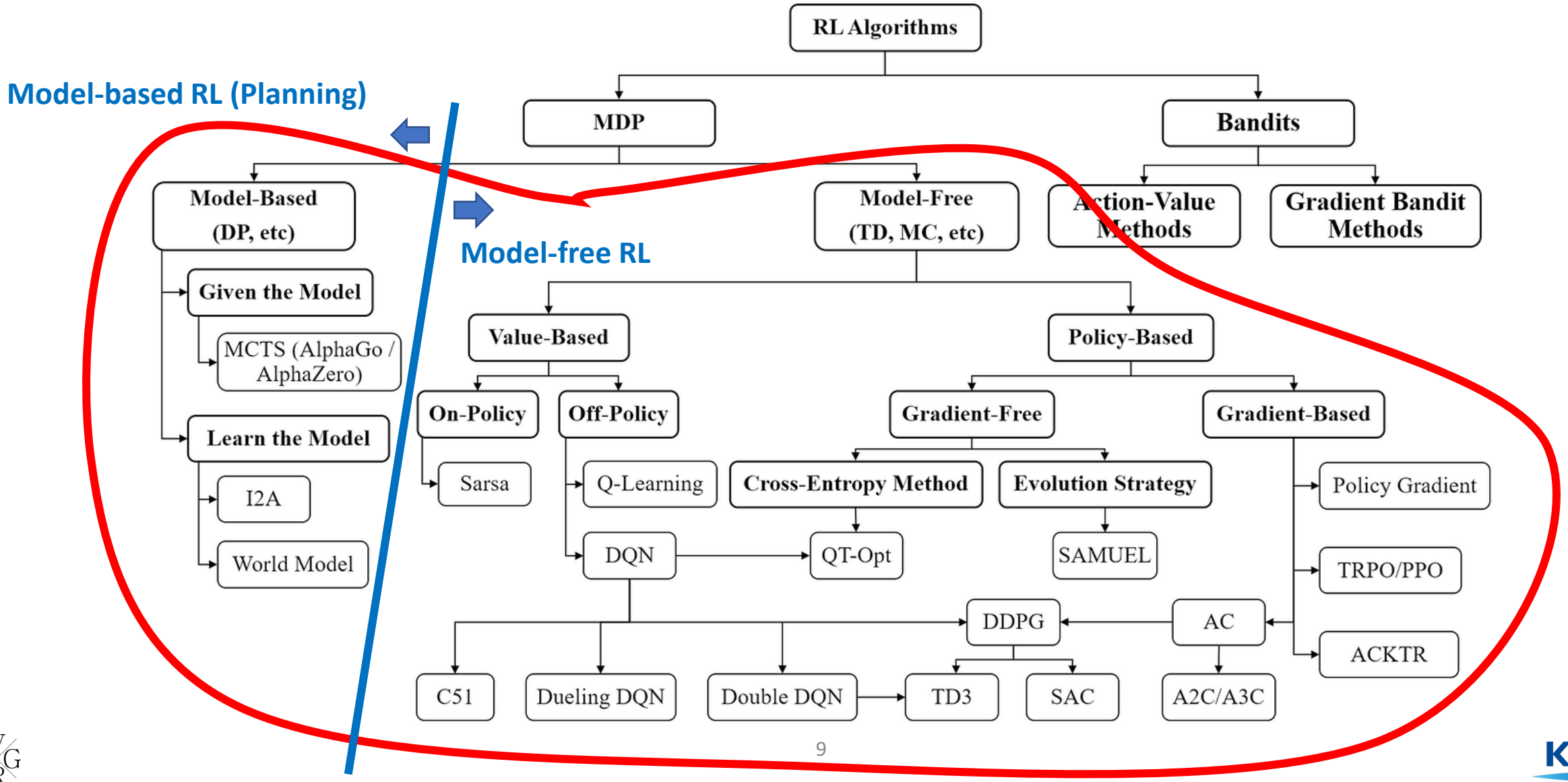
Partially Observable Markov Decision Process (POMDP)

- **POMDP** is an extension of **MDP**, where an agent cannot fully observe the entire state of the environment.
  - **States (S):**
    - All environmental states.
  - **Observations (O):**
    - Partial information the agent receives (e.g., RGB-D images).
  - **Actions (A)**
  - **Rewards (R)**
  - **Transitions (T)**
  - **Discount Factor (γ)**



- To convert POMDP into MDP,
  estimate hidden states (i.e., belief states of HMM) from a history of observations.

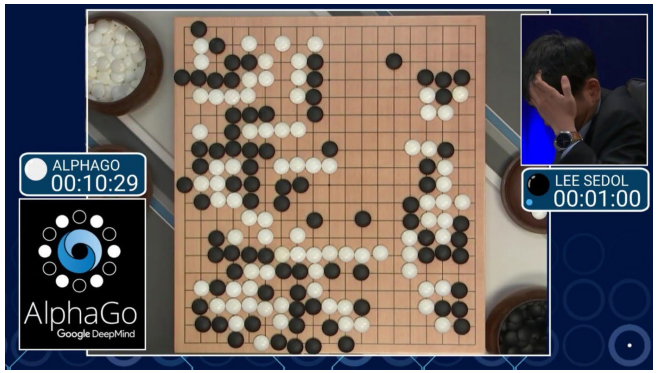
Taxonomy of RL algorithms

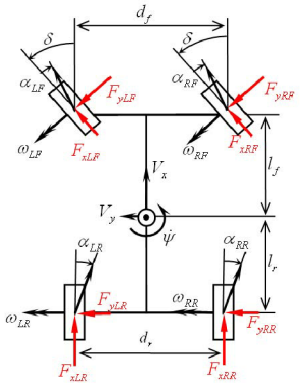


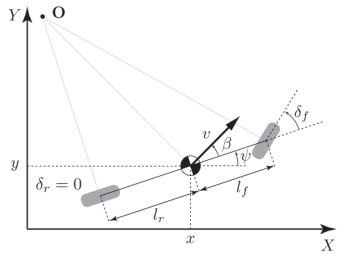
Taxonomy of RL methods

- **Model-free RL**: agent learns optimal policies directly from interactions with the environment.
  - ✅ Simplicity, Flexibility to changes in the environment
  - ❌ Poor sample efficiency, Responding to unexperienced states, Lack of planning
- **Model-based RL (Planning)**: utilizes an environment model where the agent lives

  (the model can be given naturally, or it can be learned from data)
  - ✅ Sample efficient learning, Planning capability
  - ❌ Risk of model accuracy and bias







Environment model examples

Characteristics of Reinforcement Learning

- What makes RL different from other machine learning paradigms?
  - No supervisor, only a reward signal
  - Feedback is delayed, not instantaneous
  - Time matters
    - agent's actions affect the subsequent data it will receive
    - collected data is sequential → not i.i.d. (independent and identically distributed)

Application of RL for Robot Arm Tasks (1)

ICRA LONDON 2023

Learning-based Initialization of Trajectory Optimization for Path-following Problems of Redundant Manipulators

Min-Sung Yoon, Min-Cheul Kang, Dae-Hyung Park, Sung-Eui Yoon

SVG

KAIST Korea Advanced Institute of Science and Technology

School of Computing

Problem Statement of Path-following Problems

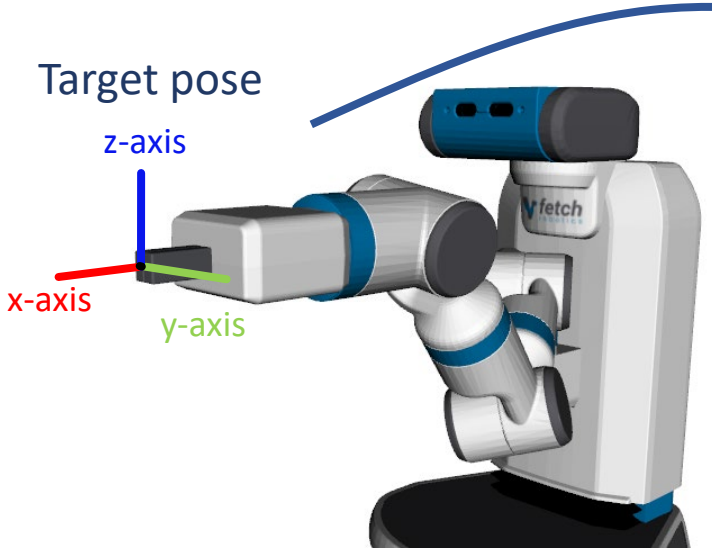
- Generate a joint trajectory precisely following a given 6-dimensional Cartesian path (i.e., target path) with an end-effector.

Target path: ‘Hello’

Target path: ‘Square’

Target path: ‘Zigzag’

Relationship with IK Problem of Path-following Problems

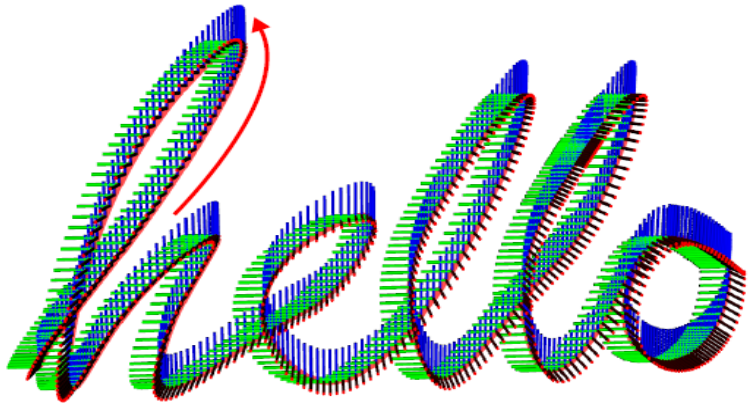


Inverse kinematics (IK) problem:

Target pose → Configuration

$\boldsymbol{x} \in \mathbb{R}^{6}$ $\boldsymbol{q} \in \mathbb{R}^{d}$

Target path: a sequence of target poses



Path-following problem:

Target path → Joint trajectory

$\boldsymbol{X = \{x_0, \dots, x_{N-1}\}}$ $\boldsymbol{\xi = \{q_0, \dots, q_{N-1}\}}$

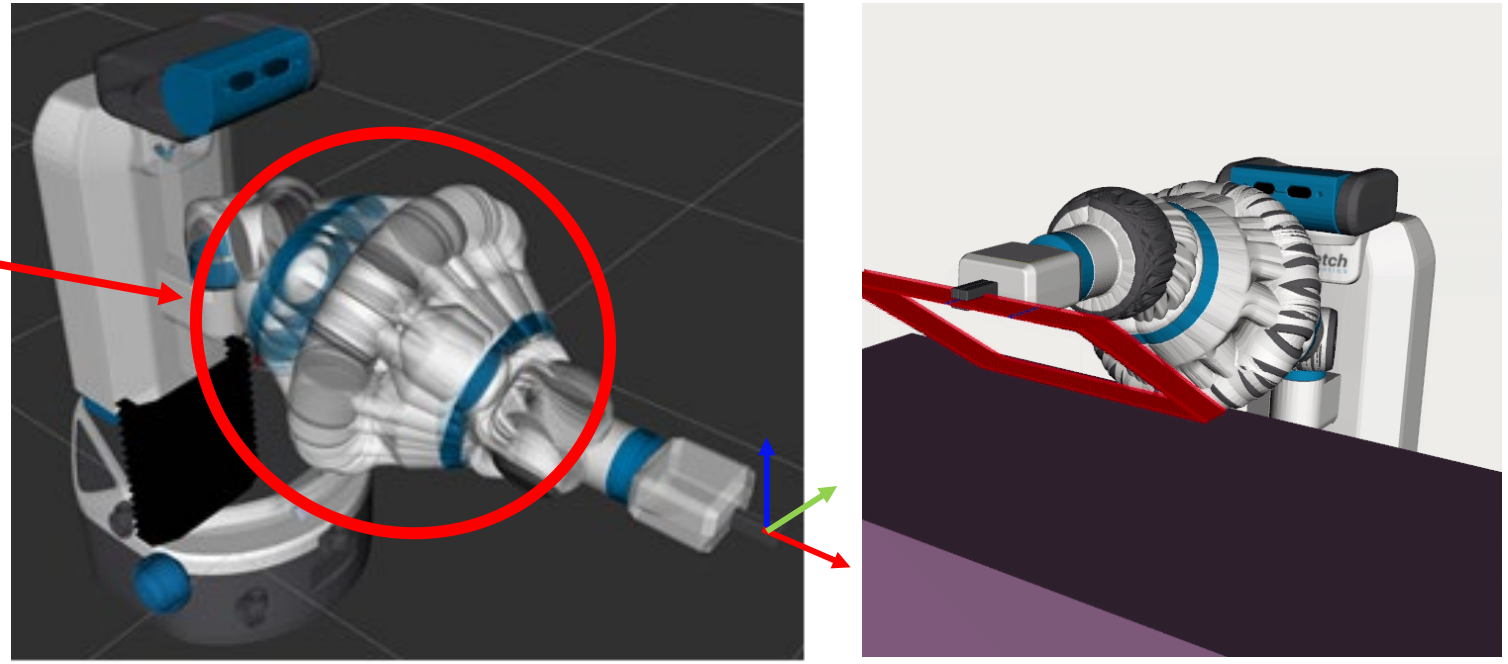
Objectives and constraints

- End-effector matching
- Collision avoidance
- …

Redundant Manipulators

- Redundant manipulators have an infinite number of Inverse kinematics (IK) solutions given a single end-effector pose.

Redundancy in joint space
→ **Redundant** motion
→ **Null-space** motion

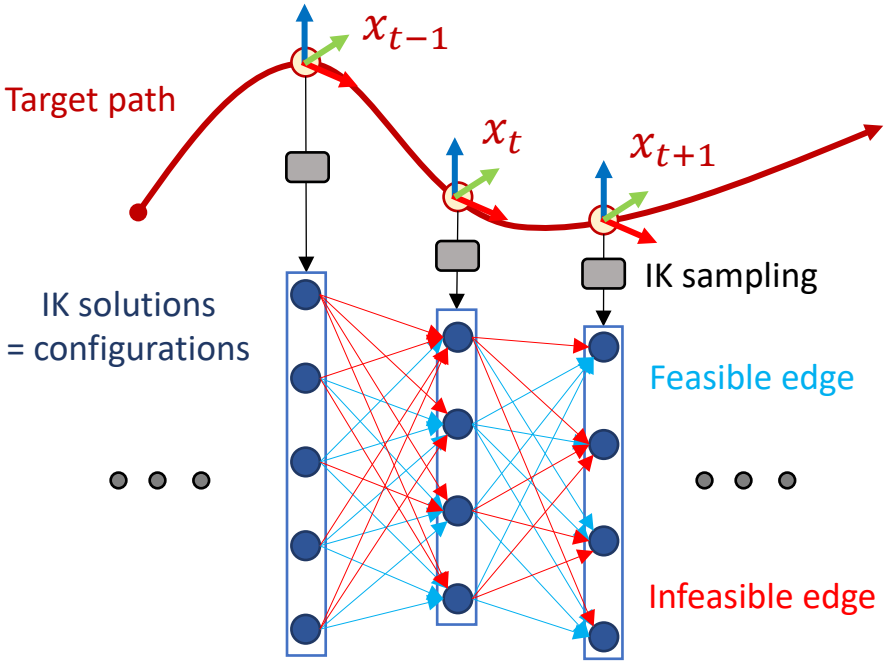


[Inverse kinematics (IK) solutions given a pose]
(Fetch robot manipulator has **7 degree of freedom**)

Approaches for Path-following Problems

Sampling-based approaches

[Holladay et al. 2019, Praveena et al. 2019, Rakita et al. 2019, …]



[Discrete configuration graph]

Asymptotically optimal

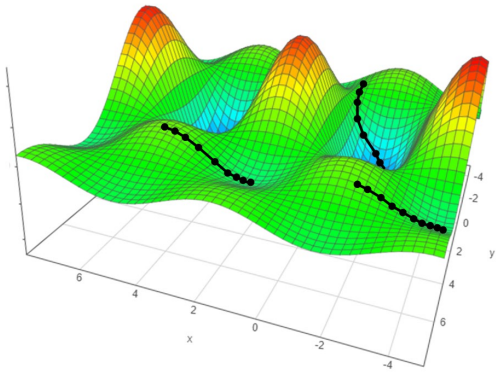
Slow convergence speed, high computational burden

Optimization-based approaches

[Kang et al. 2020, Schulman et al. 2013, Zucker et al. 2013, …]

- Variable: a whole joint trajectory, $\xi = \{q_0, \dots, q_{N-1}\}$
- Objective function:

$$\mathcal{U}(\xi) = \mathcal{F}_{pose}(\xi) + \lambda_1 \mathcal{F}_{obs}(\xi) + \lambda_2 \mathcal{F}_{smooth}(\xi),$$



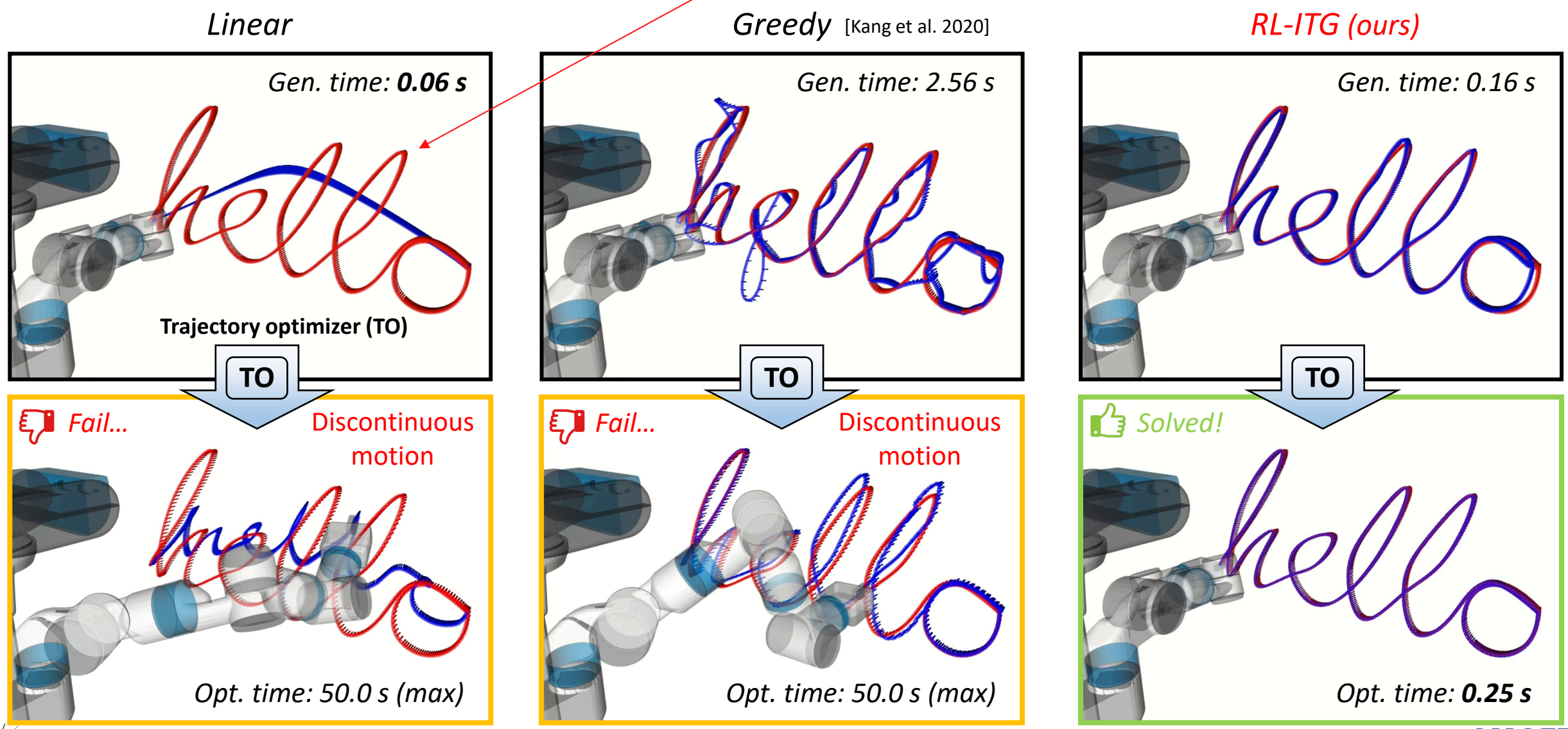
Quite fast convergence speed

Sensitive to the initial trajectory

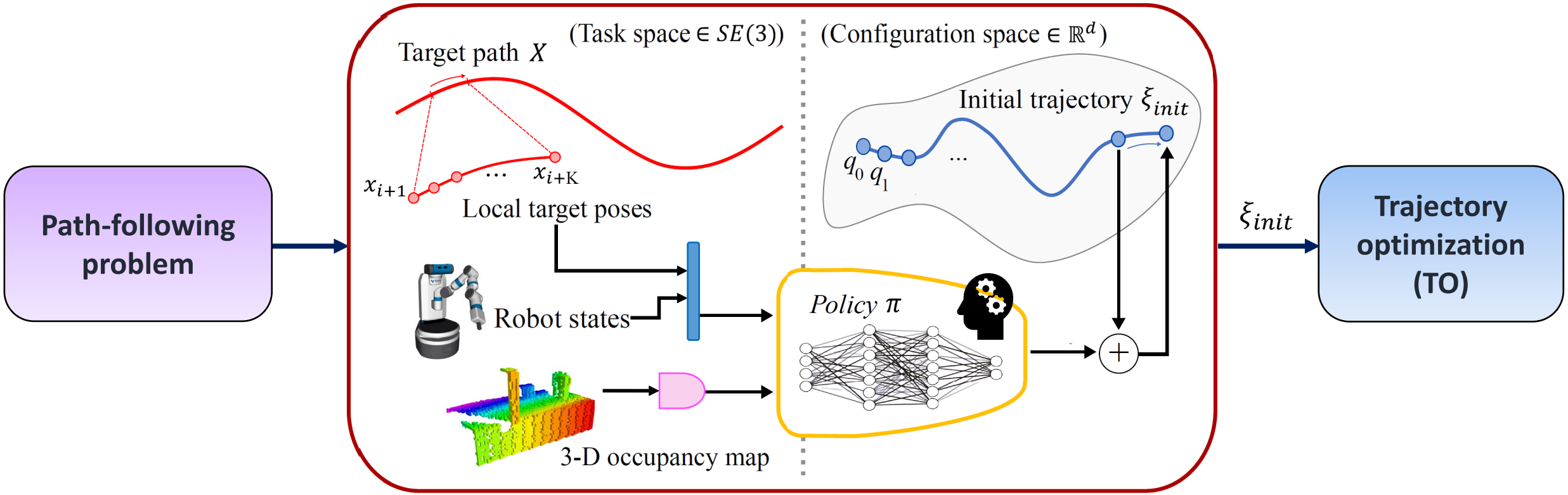
Still susceptible to poor local minima
with a practical time constraint ..

Overview

The robot must follow the target path given a fixed end-effector orientation.



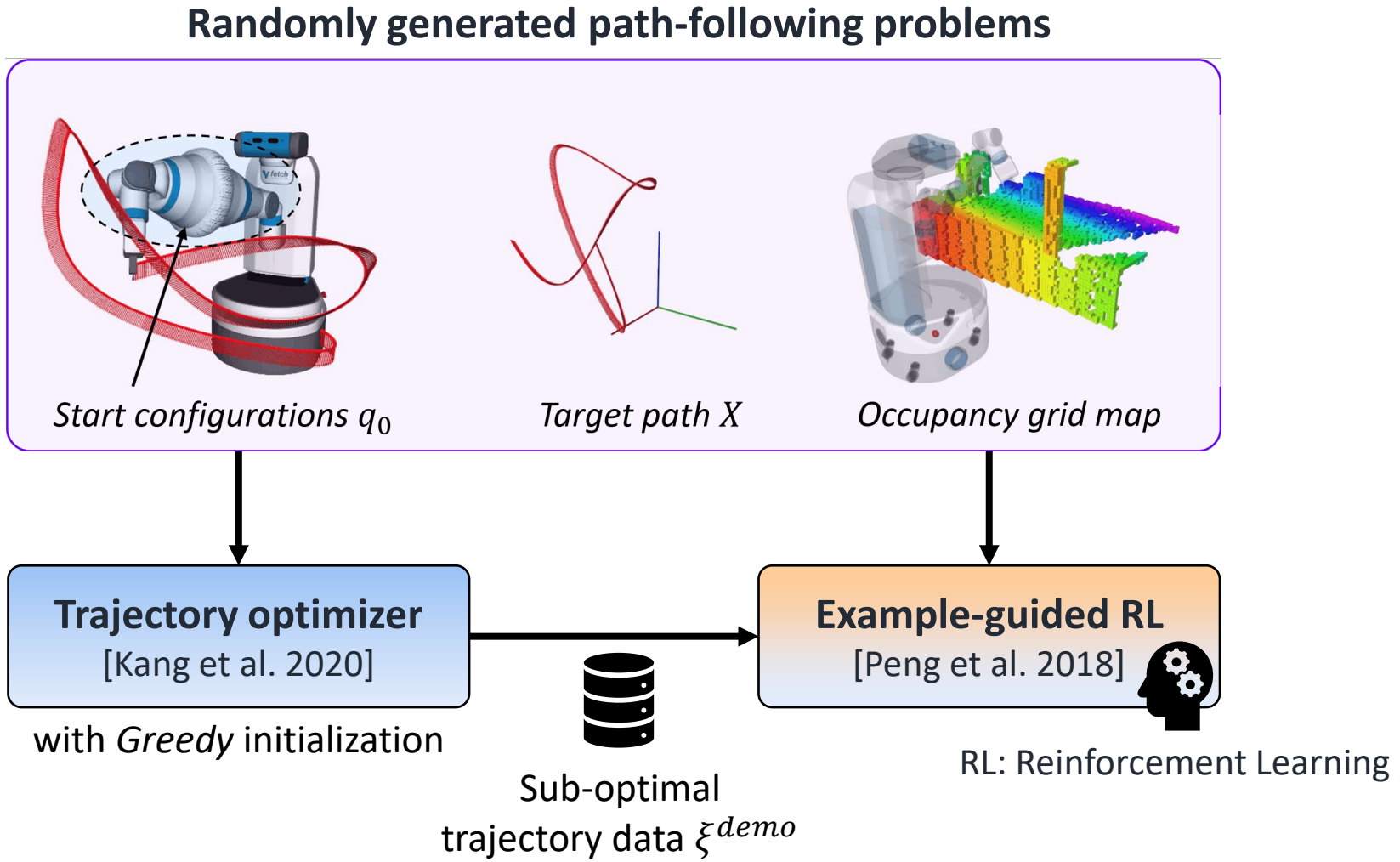
Solution: Learning-based Initial Trajectory Generator



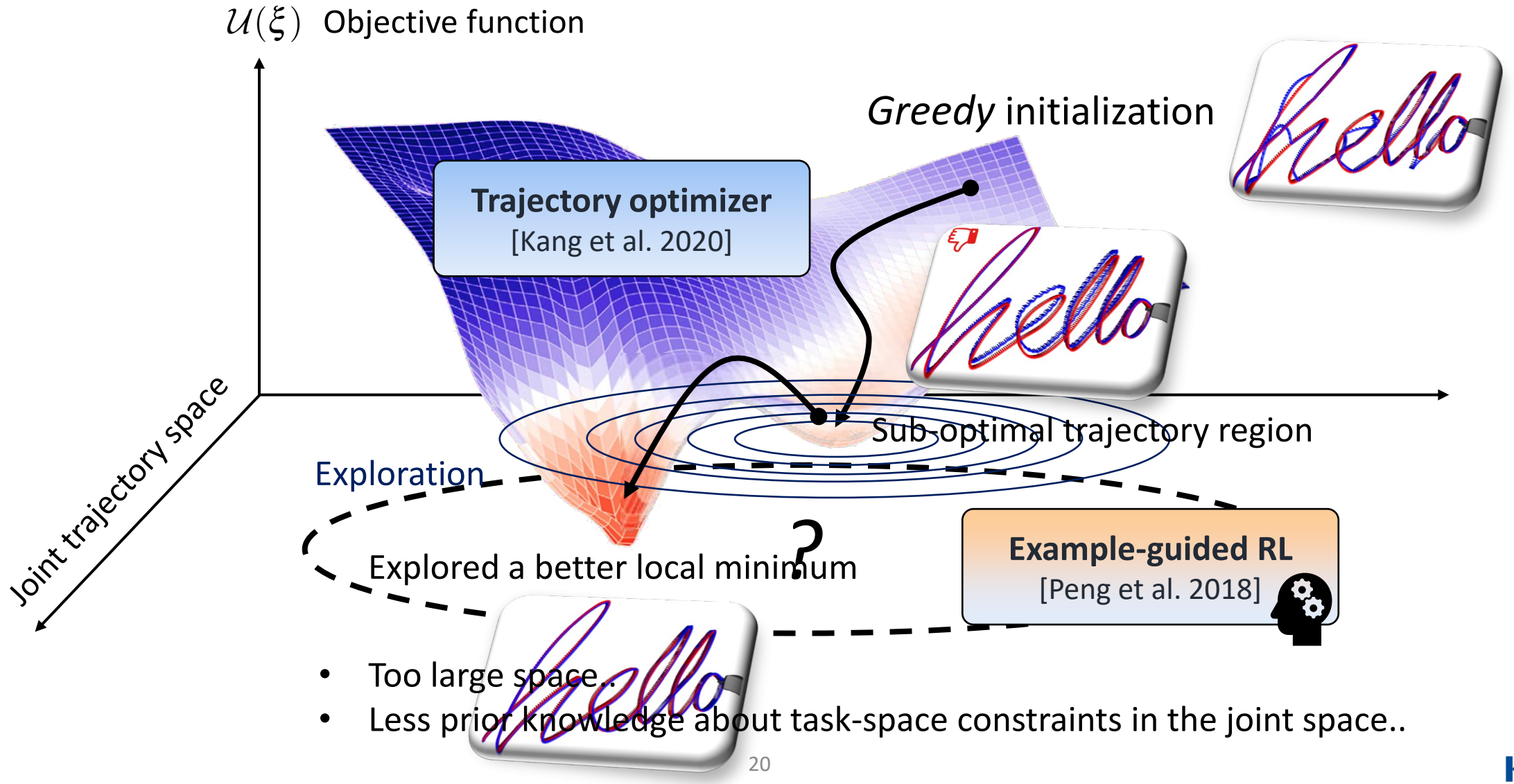
Learning-based Initial Trajectory Generator (ITG)

Solution: RL-based Initial Trajectory Generator (*RL-ITG*)

- Training Pipeline



Solution: RL-based Initial Trajectory Generator (*RL-ITG*)

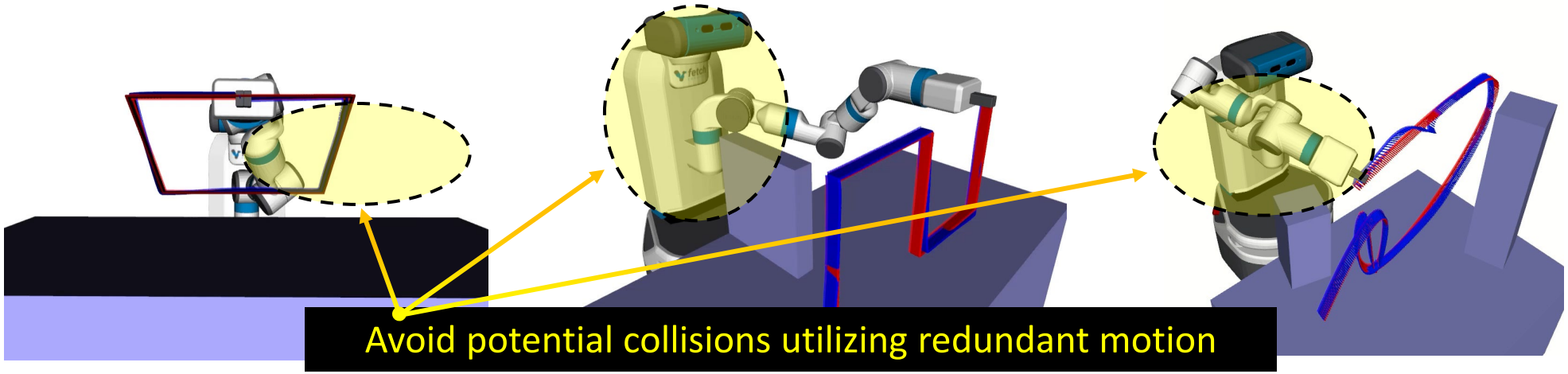


- Too large space..
- Less prior knowledge about task-space constraints in the joint space..

Solution: Initial Trajectories ξ_{init} generated by *RL-ITG*

- Benchmark:

‘Square’ ‘S’ ‘Random w/ obstacle’

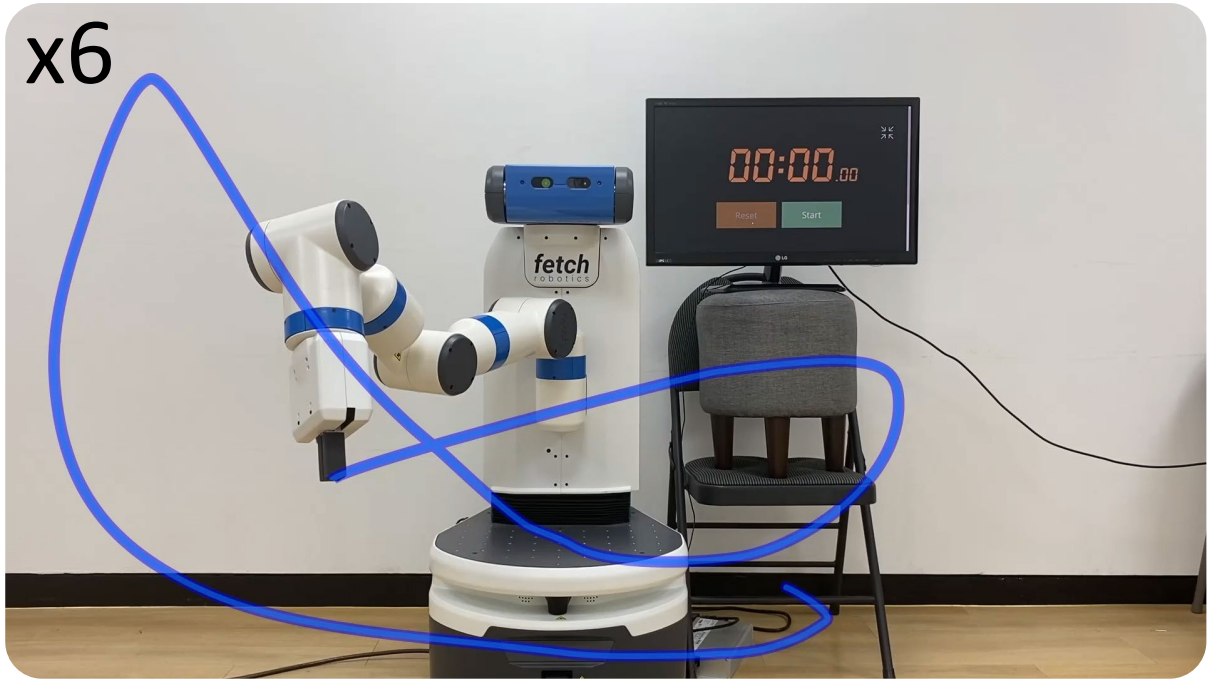


End-effector trace

Target path

‘Hello’ ‘Rotation’ ‘Zigzag’ ‘Random w/o obstacle’

Results: Real-world Experiment



- Total execution time: **00:46.76**
  - Average pose error: **2.85 x 10^{-3}**

with RL-ITG (ours)

about **220%** faster than *Greedy*

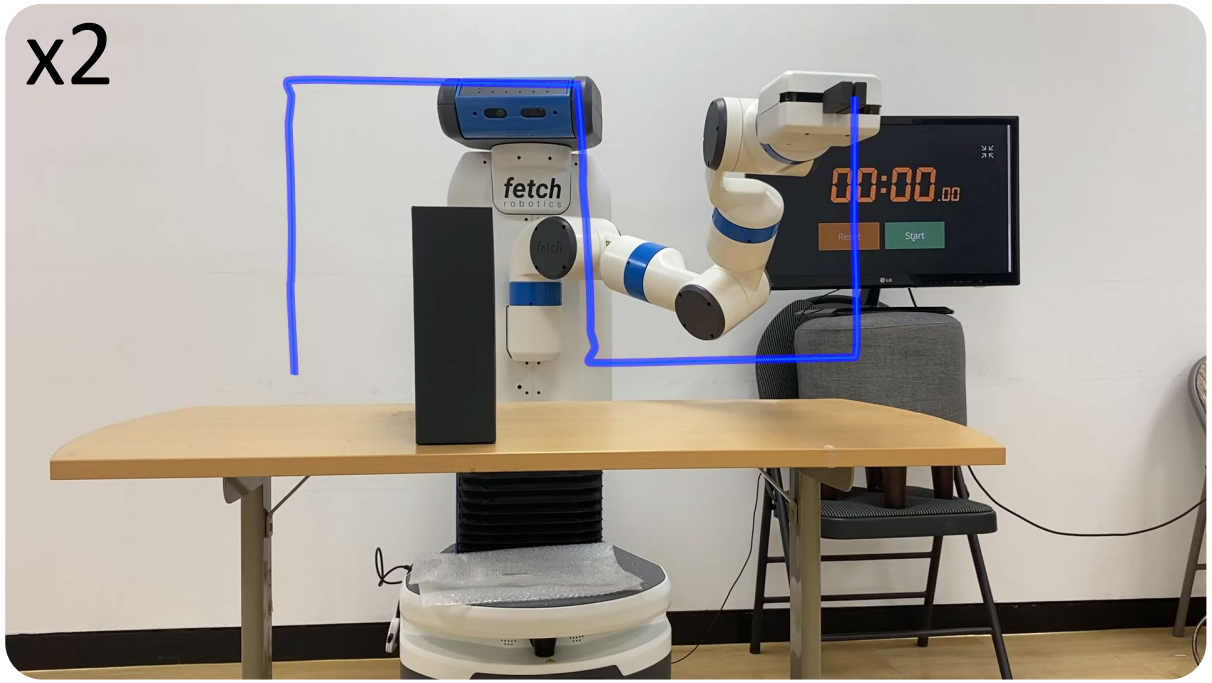
01:43.16 (min: sec)

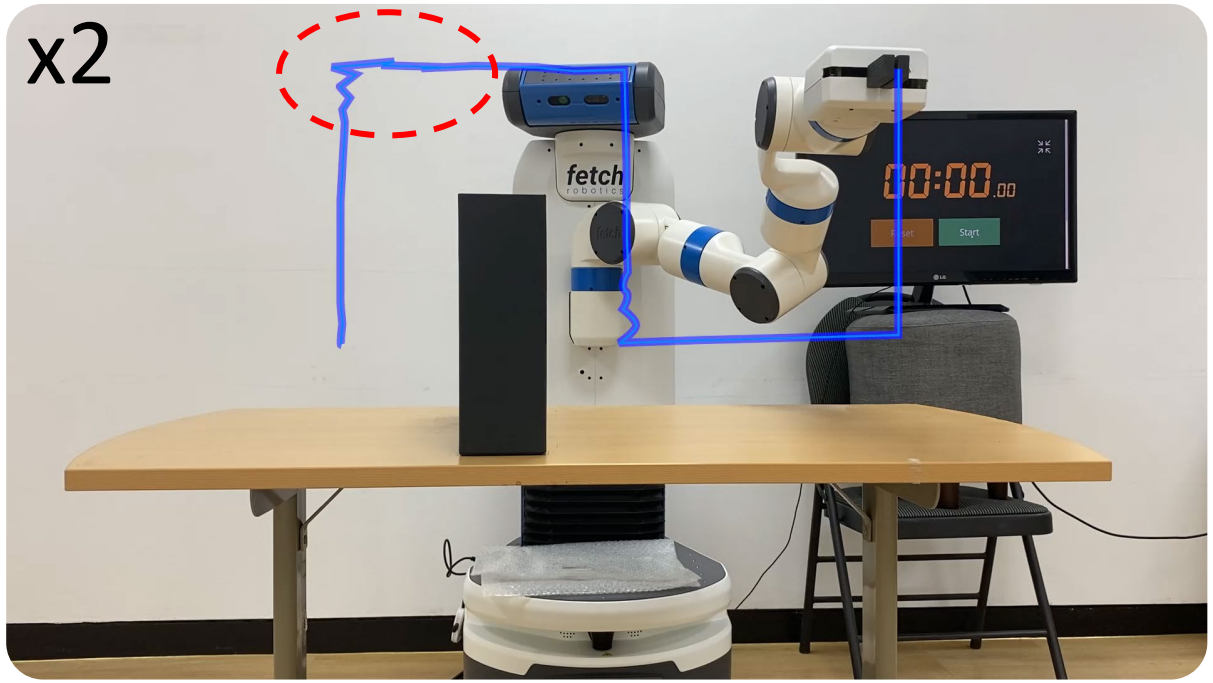
4.28 x 10^{-3}

with Greedy

Benchmark name: 'Random #64' | Trajectory optimizer: TORM [Kang et al. 2020]

Results: Real-world Experiment





- Total execution time: **00:15.23** — about **187%** faster than *Greedy* — 01:28.01 (min: sec)
  - Trajectory jerkiness: **576.729** — about **350%** smoother than *Greedy* — 2023.25 ($^{rad}/_{\sec^3}$)

with RL-ITG (ours) *with Greedy*

Benchmark name: ‘S’ | Trajectory optimizer: TORM [Kang et al. 2020]

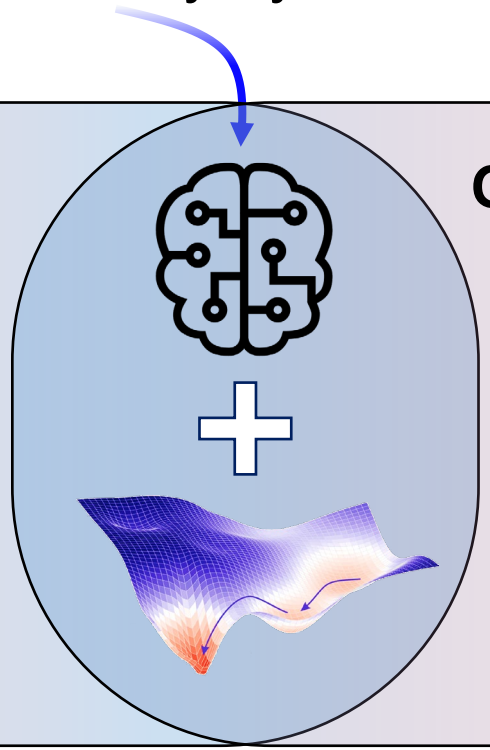
Take-Home Message

- **Hybrid frameworks integrating learning and planning** is an important strategy that works in a complementary manner.
  → Improves accuracy and efficiency by combining the two approaches.

Learning-based methods

→ may not guarantee optimality

→ but offer a good starting point for optimization quickly.



Optimization-based methods

→ may struggle with high-dimensional and non-convex problems

→ but find optimal solutions around starting point by iterative refinement.

Application of RL for Robot Arm Tasks (2)

Towards Safe Remote Manipulation: User Command Adjustment based on Risk Prediction for Dynamic Obstacles

Min-Cheul Kang, **Min-Sung Yoon**, Sung-Eui Yoon

Korea Advanced Institute of Science and Technology
School of Computing

Remote manipulation

- Performs sophisticated or hazardous tasks on behalf of humans
- Expands to irregular environments around us

Surgical robot for sophistication

Convenience store

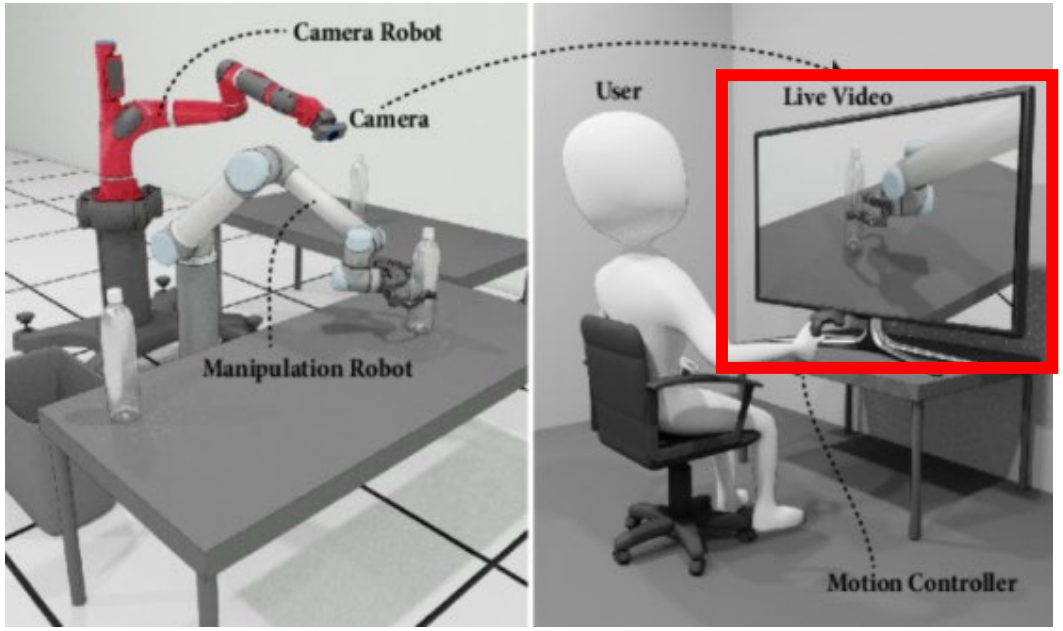
https://www.intuitive.com/
https://tx-inc.com

Motivation

- A robot accident can be a significant threat.
- A user observes a restricted environment through a camera.

  → A user may not be aware of obstacles.

Example of a robot accident

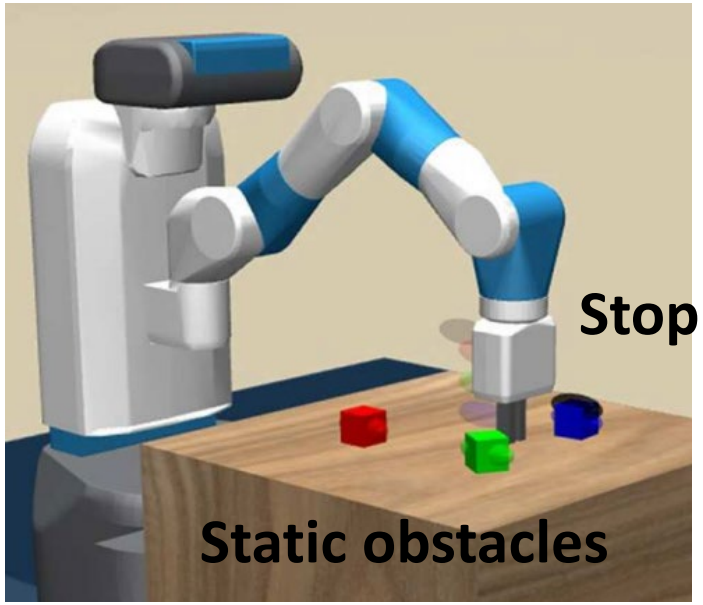


Restricted environment information

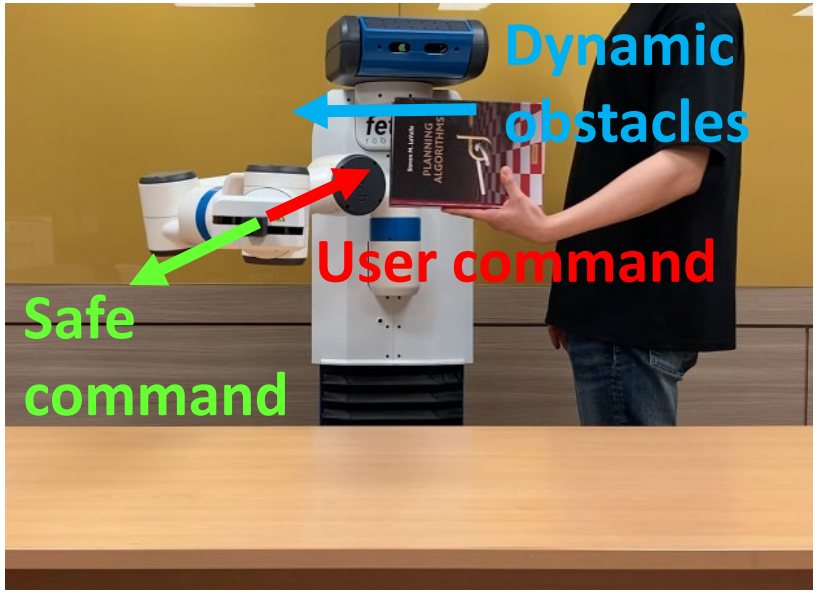
https://www.youtube.com/watch?v=mJ6I9thm-8s

Problem

- Avoiding dynamic obstacles depends on a user's judgment.
- We need a method to avoid the risk of dynamic obstacles.
  - The method should minimize the delay of remote manipulation.



Non-risky situation



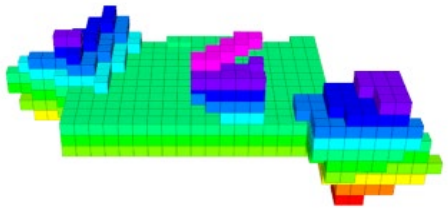
Risky situation

System flow

Δx_u: user command
Δx_π: obstacle avoidance command
$\hat{\rho}$: predicted risk for dynamic obstacles

Δx_a: adjusted command
x_a: end-effector pose for Δx_a

User command



Occupancy grid

Joint positions

Δx_u

$1 - \hat{\rho}$

Risk Prediction Network (RPN)

$\hat{\rho}$

Δx_π

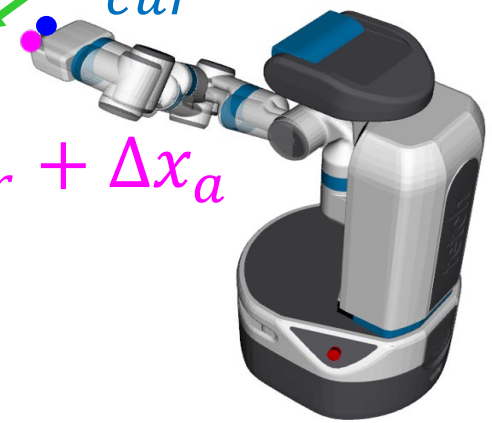
Obstacle Avoidance Command Network (OACN)

Adjusted command
$\Delta x_a = (1 - \hat{\rho})\Delta x_u + \hat{\rho}\Delta x_\pi$

Δx_a

Δx_a x_{cur}

$x_a = x_{cur} + \Delta x_a$

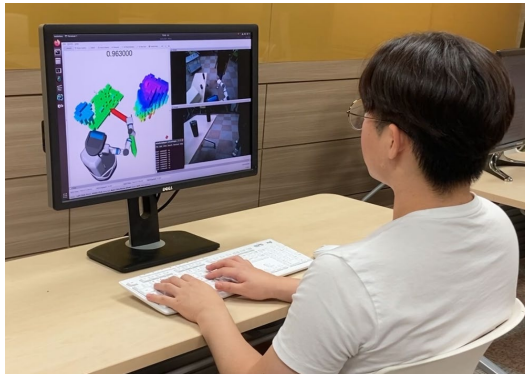


IK solver

Experimental setup

- We constructed a system for performing a remote manipulation task.

Real environment

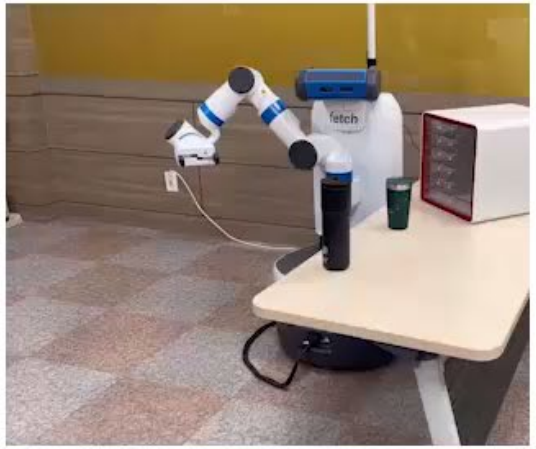


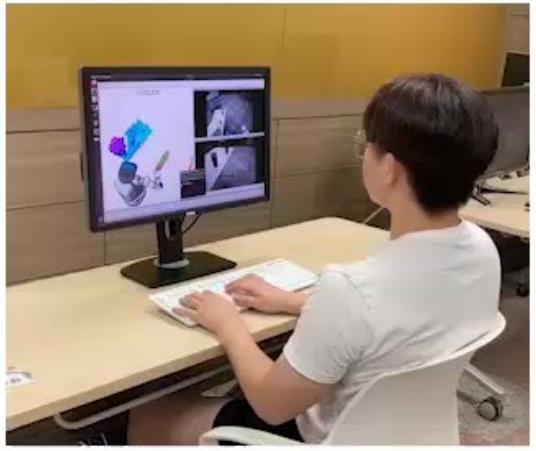
User's remote environment

User's monitor screen

Real robot experiment

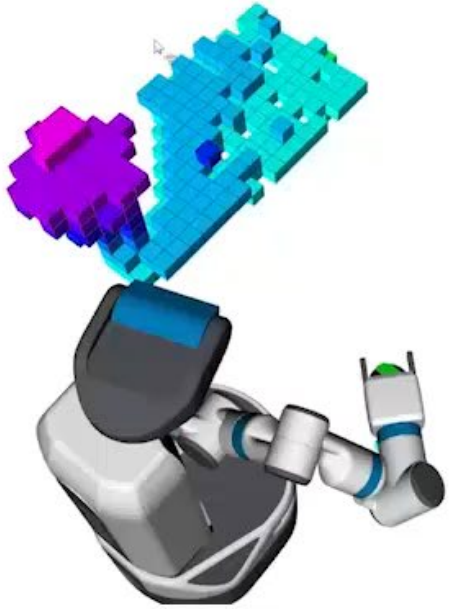
Moving to pick the water bottle

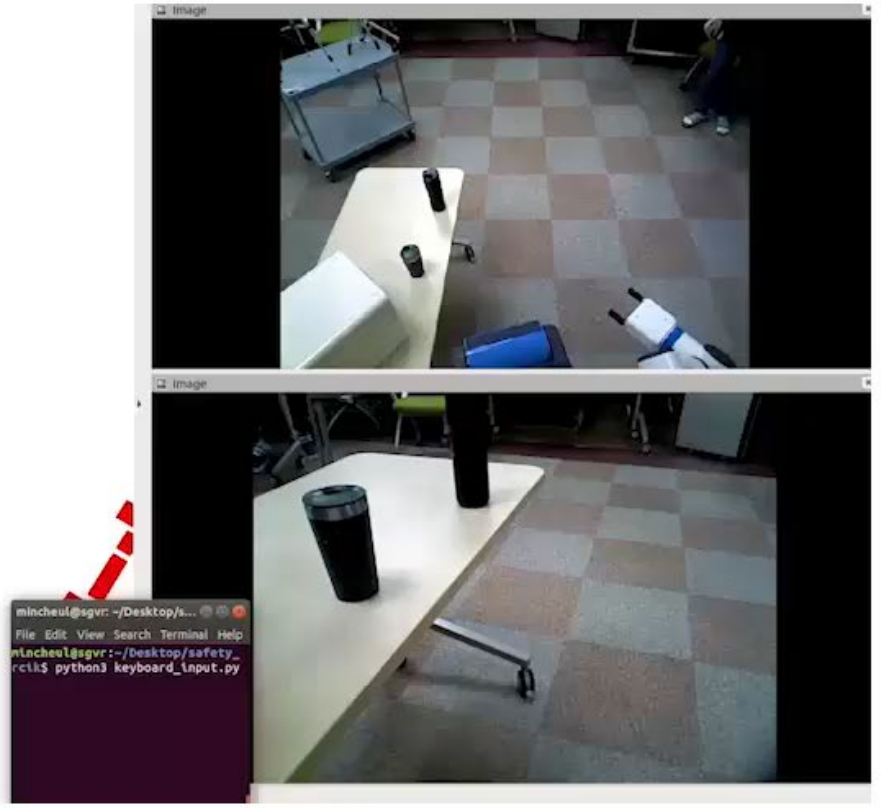




User command → Adjusted command

Risk: 0.001300



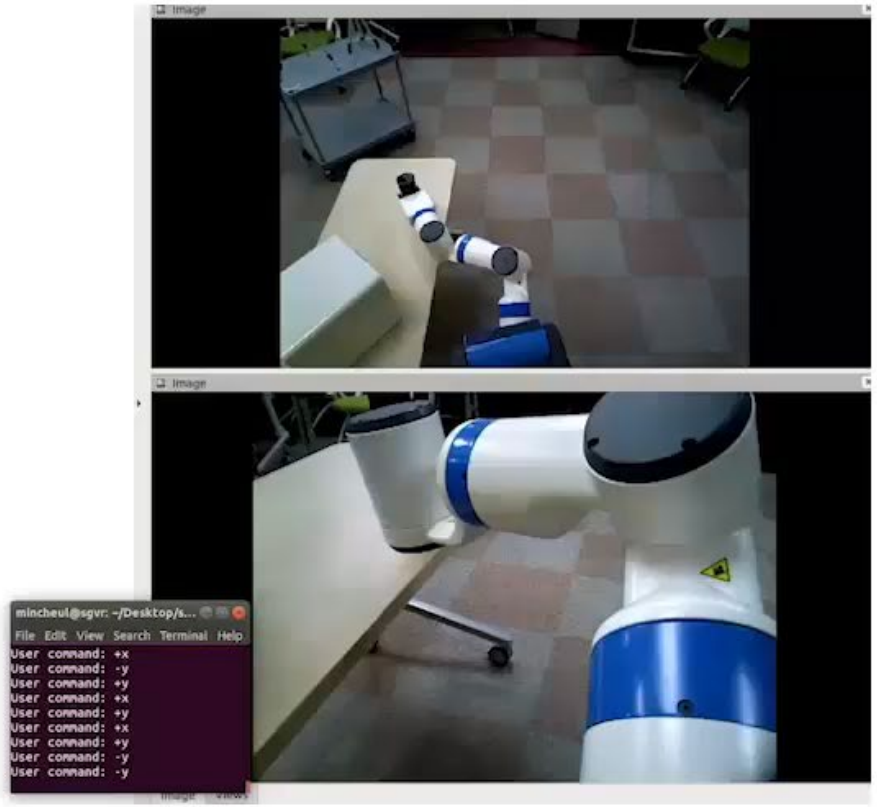


Real robot experiment

Moving to transfer the water bottle

User command Adjusted command

Risk: 0.002500



Application of RL for Quadruped robots

Enhancing Navigation Efficiency of Quadruped Robots via Leveraging Personal Transportation Platforms

Min-Sung Yoon, Sung-Eui Yoon

Learning-based Adaptive Control of Quadruped Robots for Active Stabilization on Moving Platforms

Min-Sung Yoon, Heechan Shin, Jeil Jeong, Sung-Eui Yoon

Korea Advanced Institute of Science and Technology

School of Computing

Recent Progress of Quadruped Robots

(Xuxin Cheng, et al. 2024)


(Tairan He , et al. 2024)

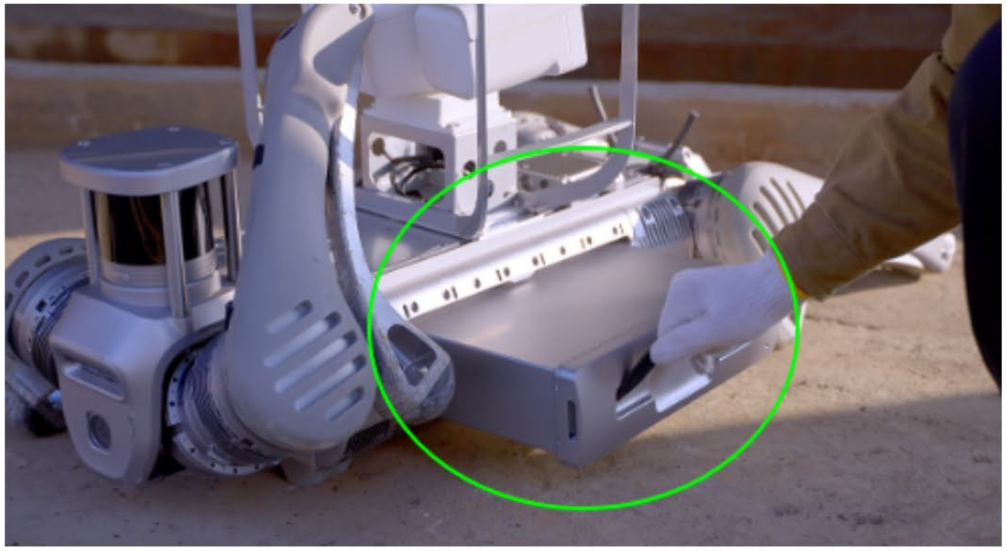

(Choi, Suyoung, et al. 2023)

(Shin, Young-Ha , et al. 2023)

Traversing Challenging Terrains

Agile Locomotion

Battery Limitation of Legged Robots



Unitree B2's Battery Spec.

- Battery Life: **4 ~ 6 hours**
  - Battery Capacity: **45Ah (2250 Wh)**
  - Standard Voltage: **50.4V**

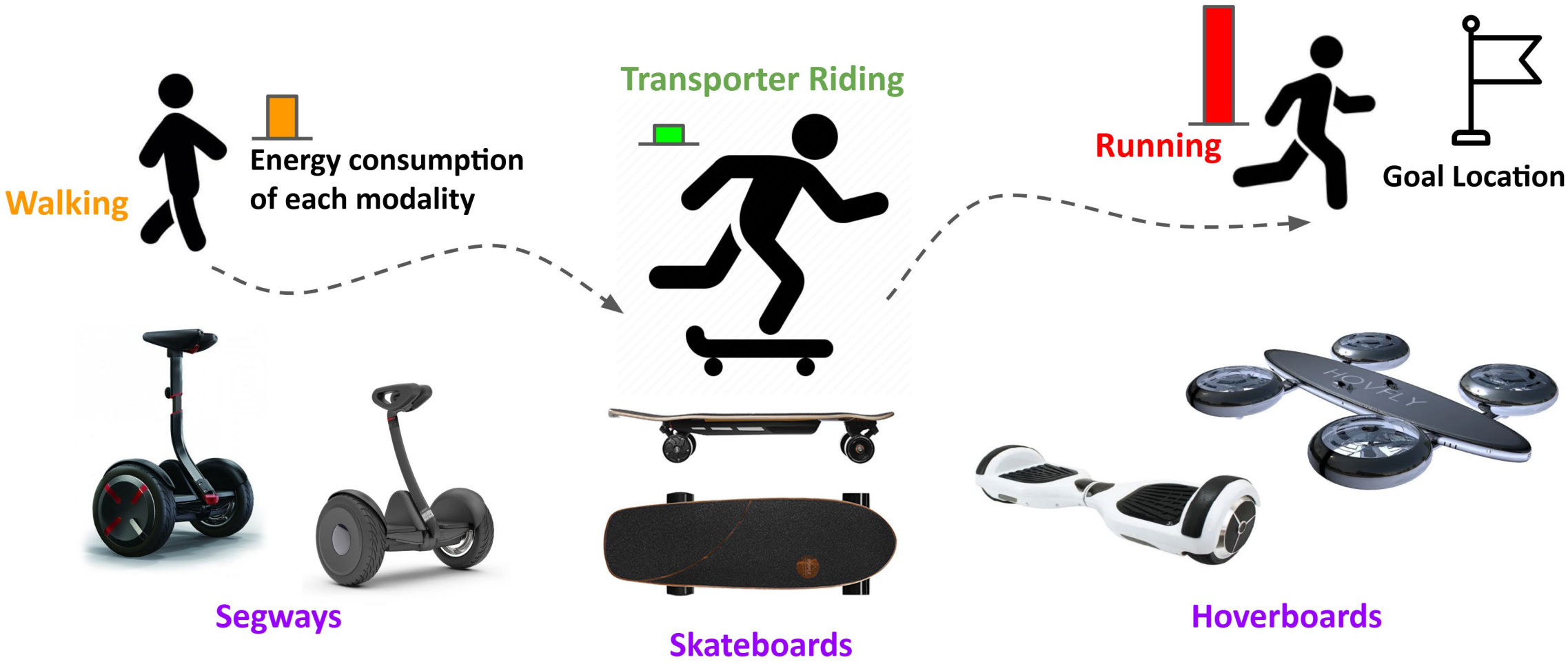
Unitree Go1's Battery Spec.

- Battery Life: **~ 2.5 hours**
  - Battery Capacity, **6Ah (133.2Wh)**
  - Standard Voltage: **22.2V**

Actual running time is way more below…

Motivation: Human Mobility Augmentation

- Humans Use **Transporters** to Move Farther and Faster.



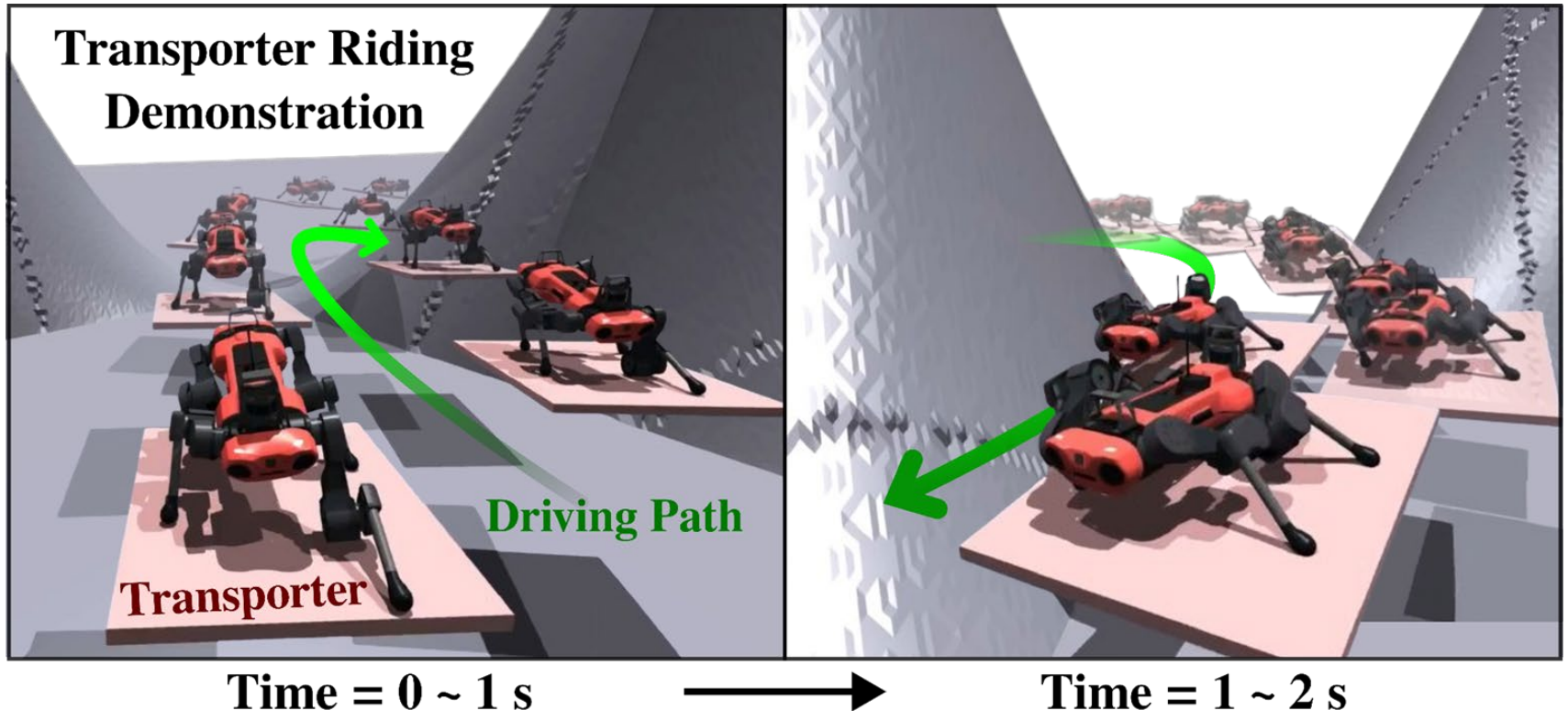
Beyond Footsteps: Transporter-Riding Skills

- Legged animals, like dogs, instinctively use transporters to improve mobility and reduce energy use.



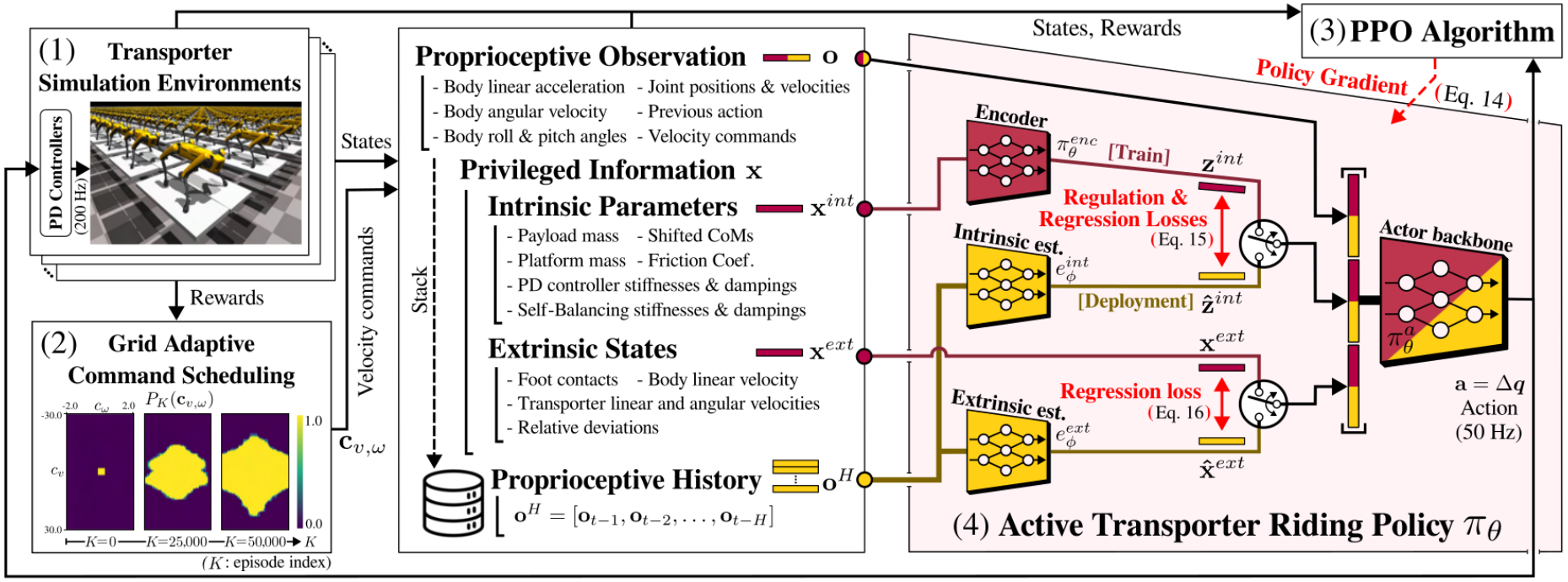
Research Goal

- We aim to ensure that quadruped robots adeptly utilize transportation platforms, also known as transporters, for efficient long-range navigation.

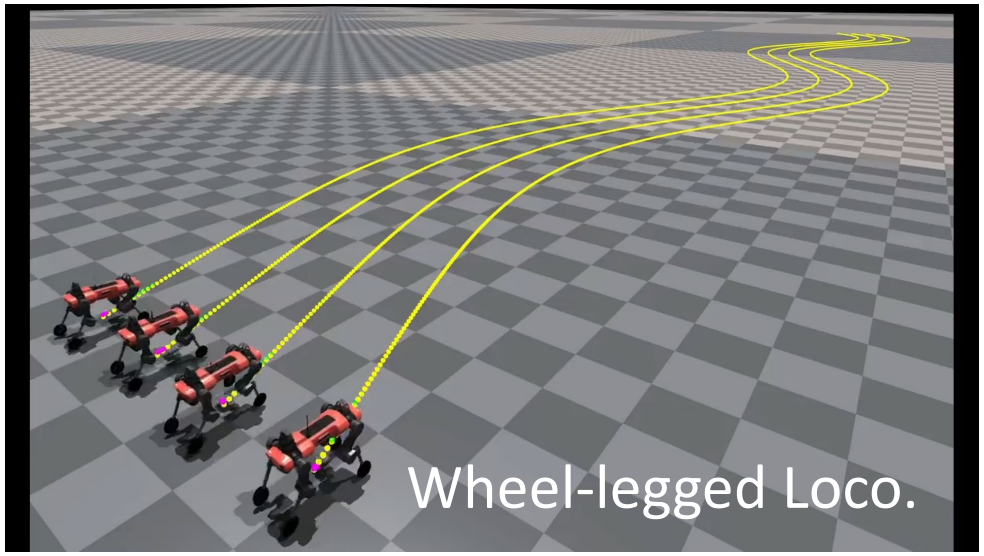


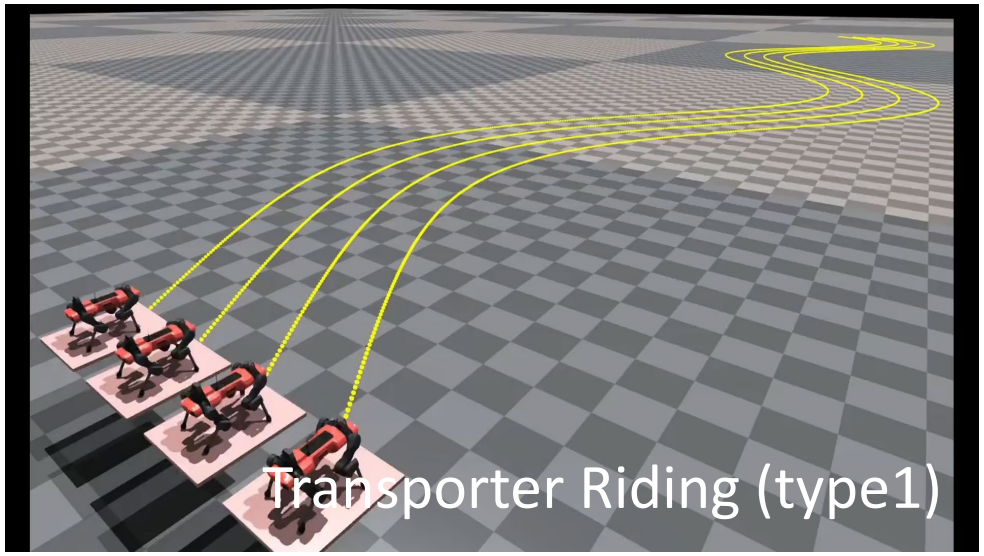
Main Contribution

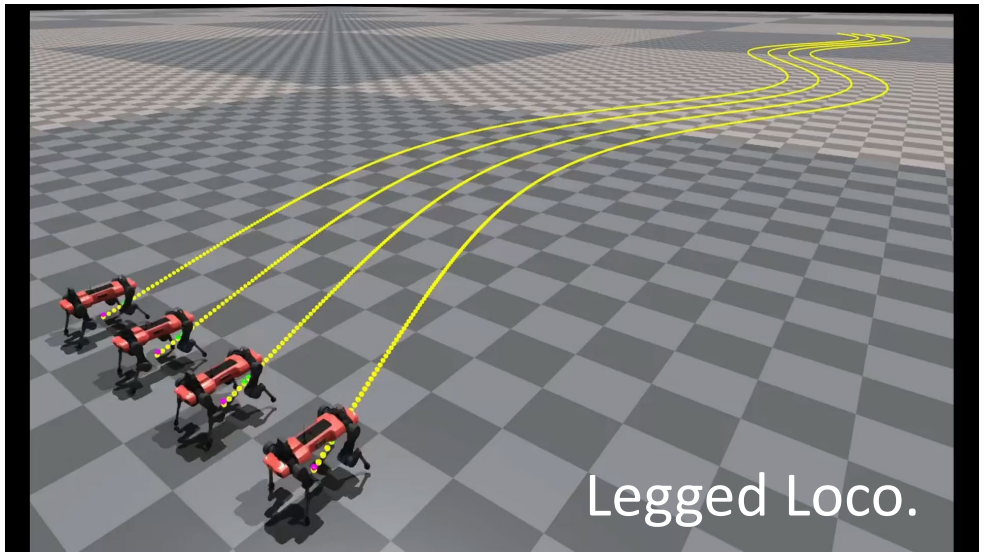
- We introduce RL-ATR (Reinforcement Learning-based Active Transporter Riding method).
  - Built a simulation with transporter dynamics for reinforcement learning.
  - Trained a transporter riding policy.
  - Added state estimators for stability in non-inertial frames (moving platforms).

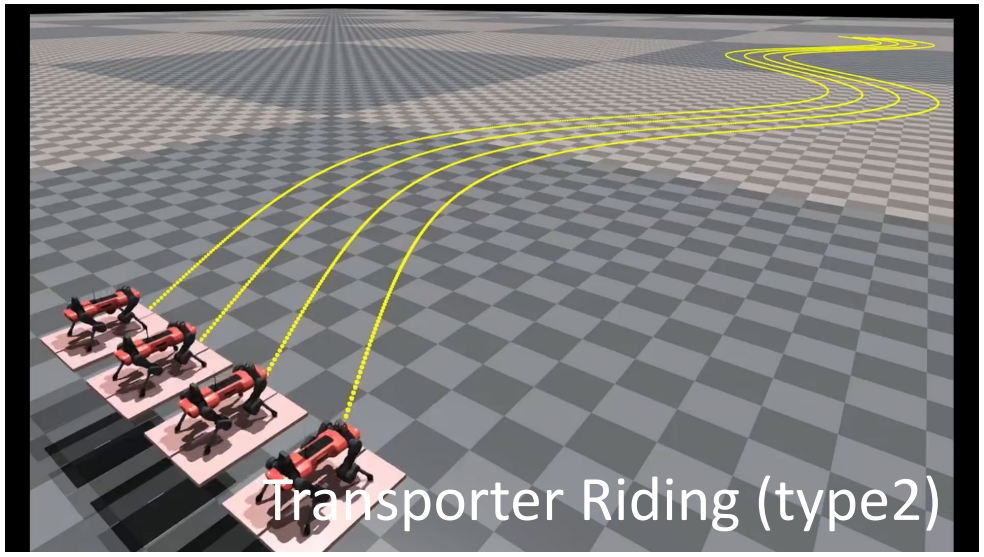


Experimental Result



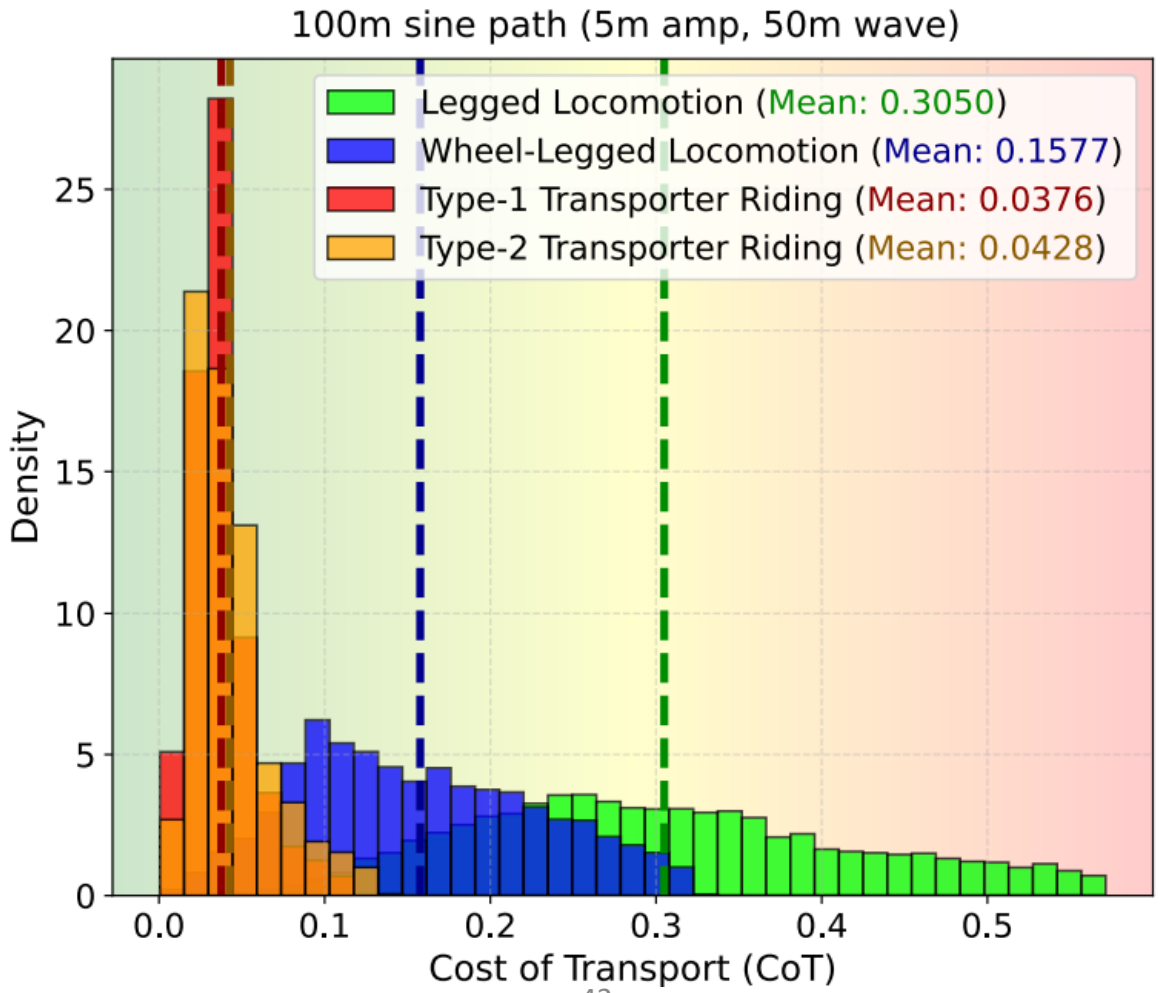






Experimental Result

Distribution of Cost of Transport (CoT) values measured during the path tracking tasks



Summary

- Covered basics of RL and its applications to robot arms and quadrupeds.
- RL enables diverse real-world tasks, showing broad applicability.
- Reward engineering is often needed to guide desired behavior.
- Despite these challenges, RL offers unique and powerful capabilities.

Thanks

Any Questions?

Korea Advanced Institute of Science and Technology
School of Computing