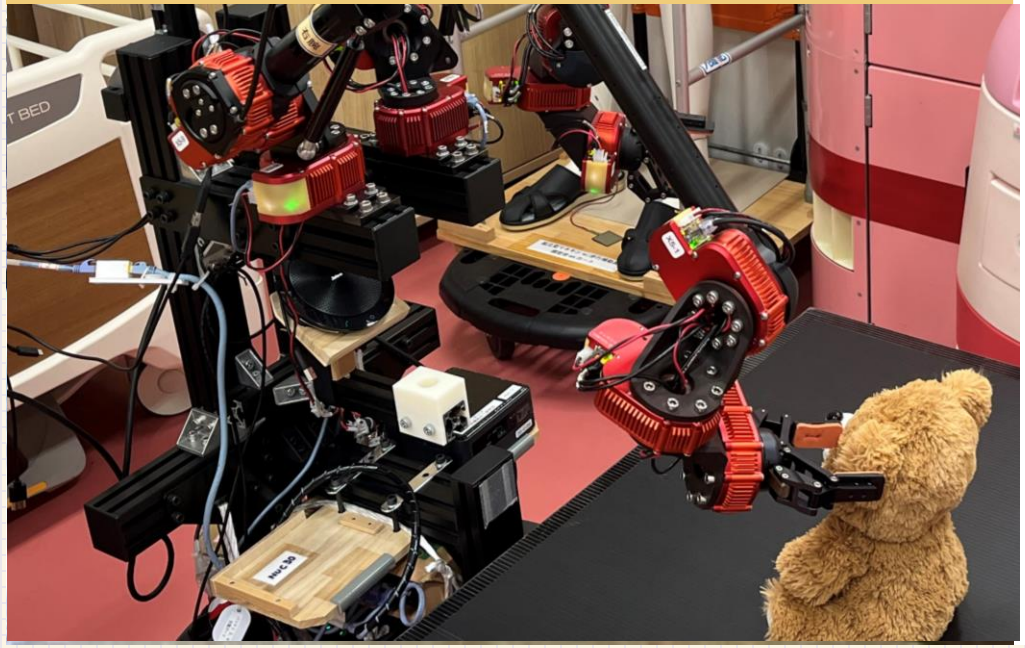
KAIT International Program
Final Presentation

July 24, 2025, 16:00~19:00 JST
K1 Bldg, Room 1201
Kanagawa Institute of Technology, Japan



Rev.2

A Tool Manipulation System to Predict the Effect of Behaviors based on Efference Copies

Sansak Lowmankong, Pimolkan Piankitrungreang [1)]
Haruki Isono, Asumu Sato, Shota Arai, Ryo Saegusa [2)]
1) Mechanical Engineering, Chulalongkorn University
2) Information Technology, Kanagawa Institute of Technology

Recap of the objectives

- Meet Mani, a robot with 2 arms consisting of 7 DOFs in each. It has been programmed to serve food and drinks directly to people's mouths and hands, even holding cutlery with its gripper.
- Mani’s gripper is capable of applying various tools usage, which is the advantage of the 7 DOFs arms that could be used in many high-precision actions.
- We will develop algorithm using real-time data to adjust the robot's actions, depending on its environment and task.
- In this research, we had implemented something called "Efference Copy" into our robot to help it use tools.



[1]

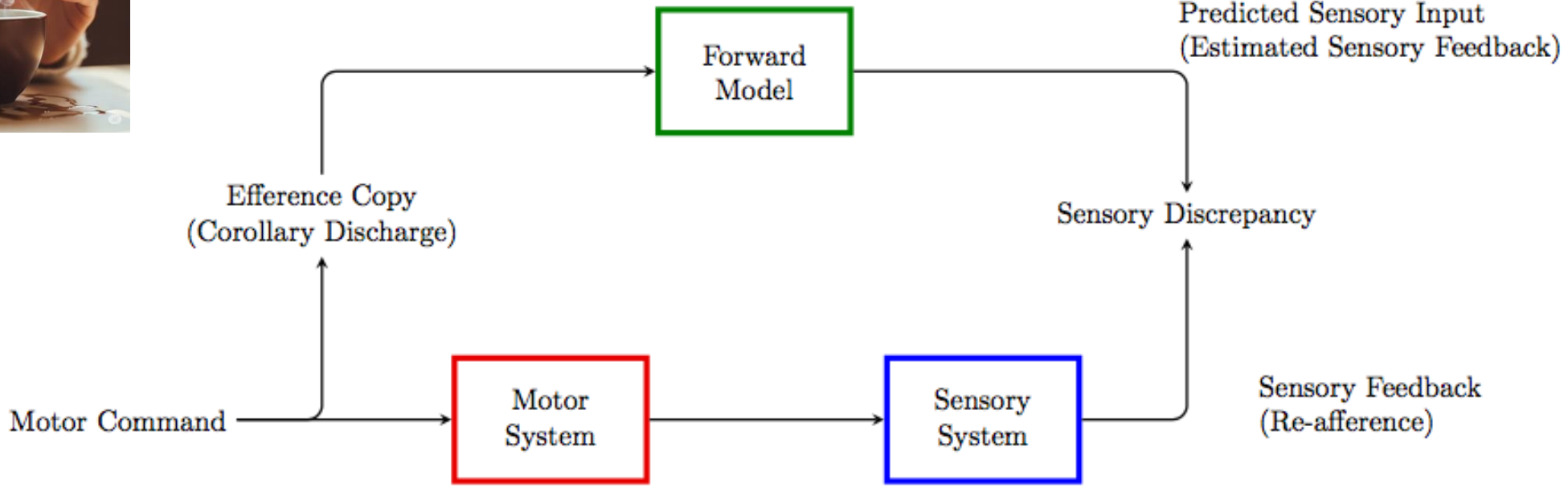


Proposed Method(1)



[AI]

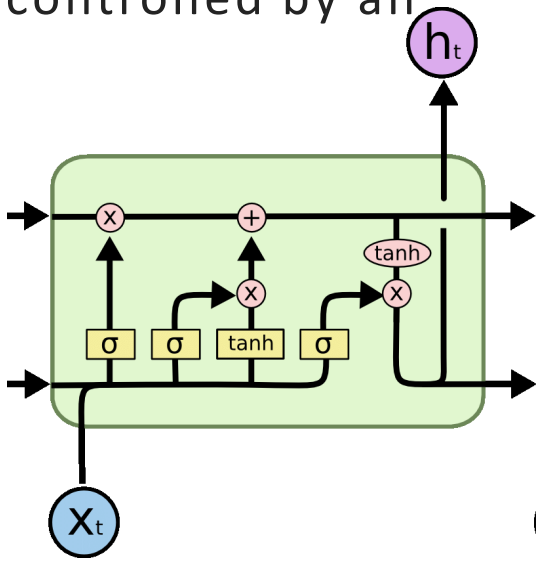
Efference Copy



[2]

Proposed Method(2)

- **LSTMs** are advanced Recurrent Neural Networks (RNNs) that excel at processing and remembering information in **sequential data** over long periods. They overcome limitations of traditional RNNs by using internal "gates" that regulate the flow of information into and out of a "cell state" (memory), making them **ideal for robot perception and control involving time-series data**.
- LSTMs control memory flow using gates. The **Cell State (c_t)** is the memory, updated by a **Forget Gate (f_t)** and an **Input Gate (i_t)**, and its output is controlled by an **Output Gate (o_t)**.
- Cell State Update: $c_t = f_t \cdot c_{t-1} + i_t \cdot \tilde{c}_t$
- Hidden State Output: $h_t = o_t \cdot \tanh(c_t)$

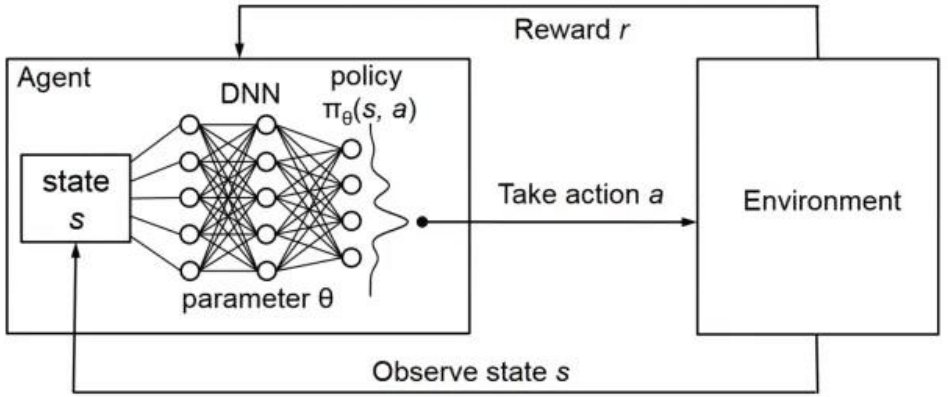


[3]

Proposed Method(3)

- **DQN** is a **Reinforcement Learning** algorithm enabling robots to learn optimal actions through trial and error. It uses a **deep neural network** to approximate the **Q-function** (expected future reward for actions). Key techniques like **experience replay** and a **target network** stabilize learning, allowing robots to master tasks in complex environments with high-dimensional sensor data.

[4]



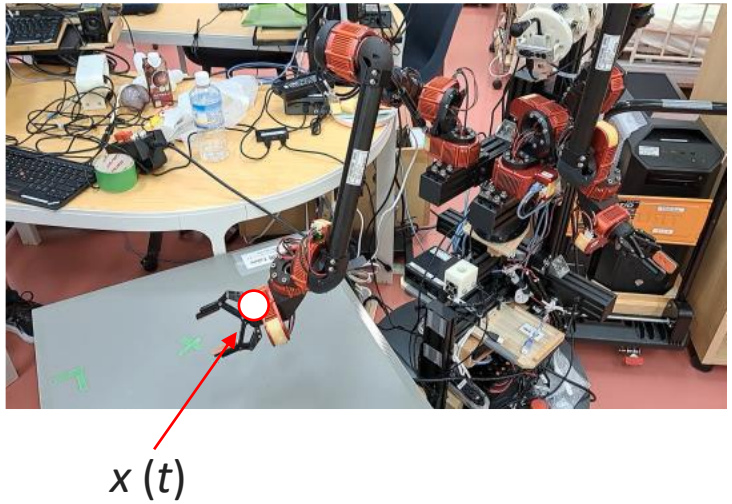
- Target Q-Value (from Bellman Equation):

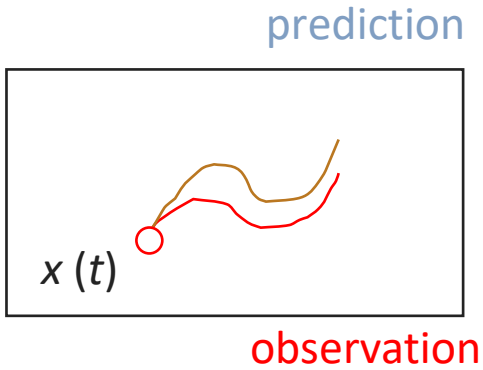
$$Y_t = r_{t+1} + \gamma max_{a'} Q(s_{t+1}, a'; \theta_{target})$$

  - r_{t+1} : Immediate reward
  - γ : Discount factor
  - θ_{target} : Stable target network weights

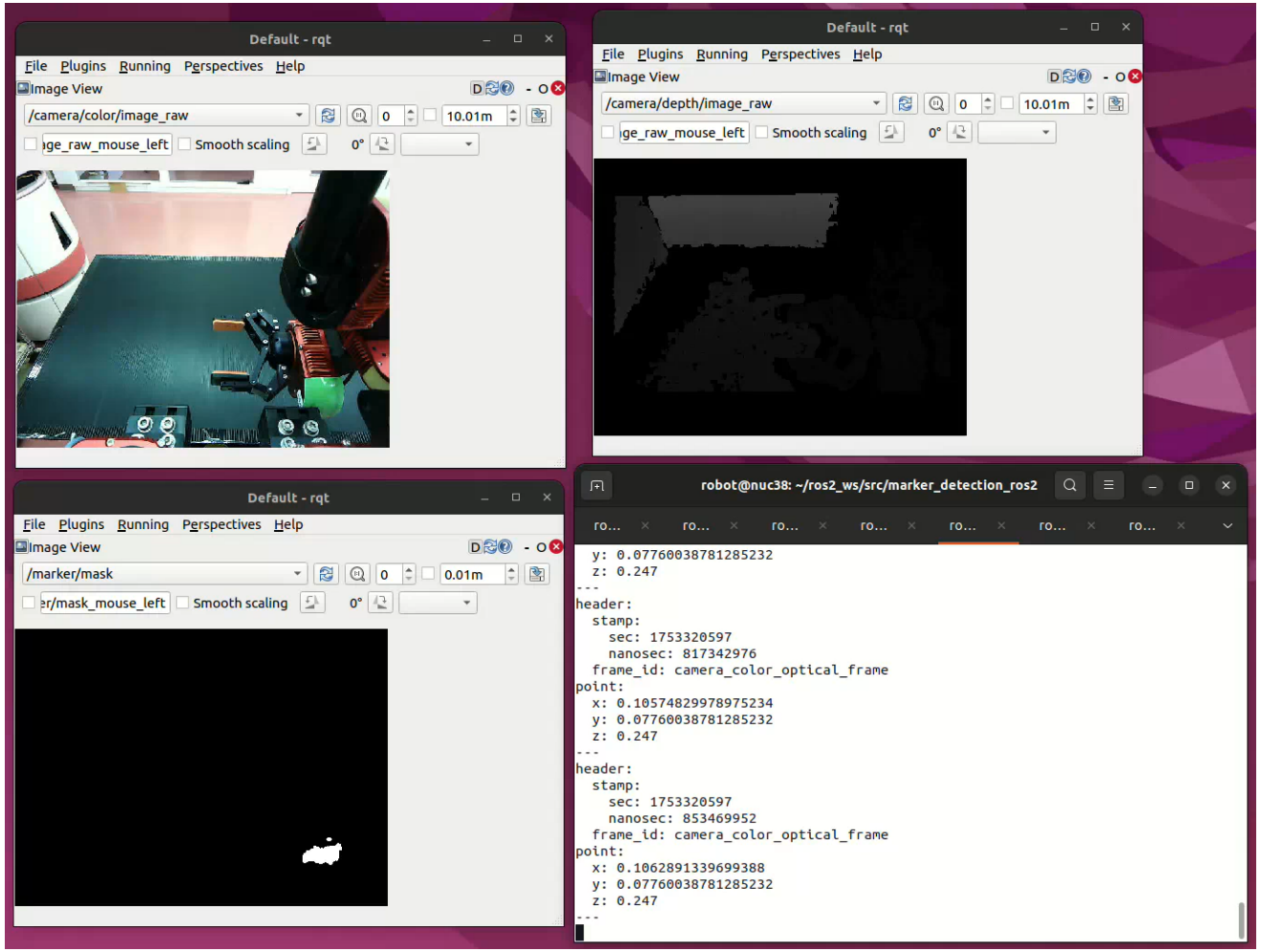
Action Perception by Efference Copies

- Action Target
  - Arm Trajectory: T_{NL}, T_{NC} , T_{NR} , T_{GL}, T_{GC} , T_{GR}
    - N: Naked arm movement, G: Grasping arm movement
    - L: Left trajectory, C: Center trajectory, R: Right trajectory
- Sensing and Motor Command
  - Arm View: I_m
  - Tracking Point: x_{th}, x_{sh}, x_{to}, x_{so}
    - T: Target of tracking point, S: Detected tracking point
    - A: Tracking point on the Hand, O: Tracking point on an Object
  - Joint Motor Position: u_c, u_e
    - C: motor command, E: motor encoder
- Learning and Testing
  - Motor Generation: $u_c(t+\Delta t) = F(x_t(t), x_s(t))$
  - Sensory Prediction: $x_s\hat{}(t+\Delta t) = G(x_s(t), u_c(t))$
    - Learning F and G, where $\{x_t(t_i)\}_{i=1,\ldots,n}$ is given.

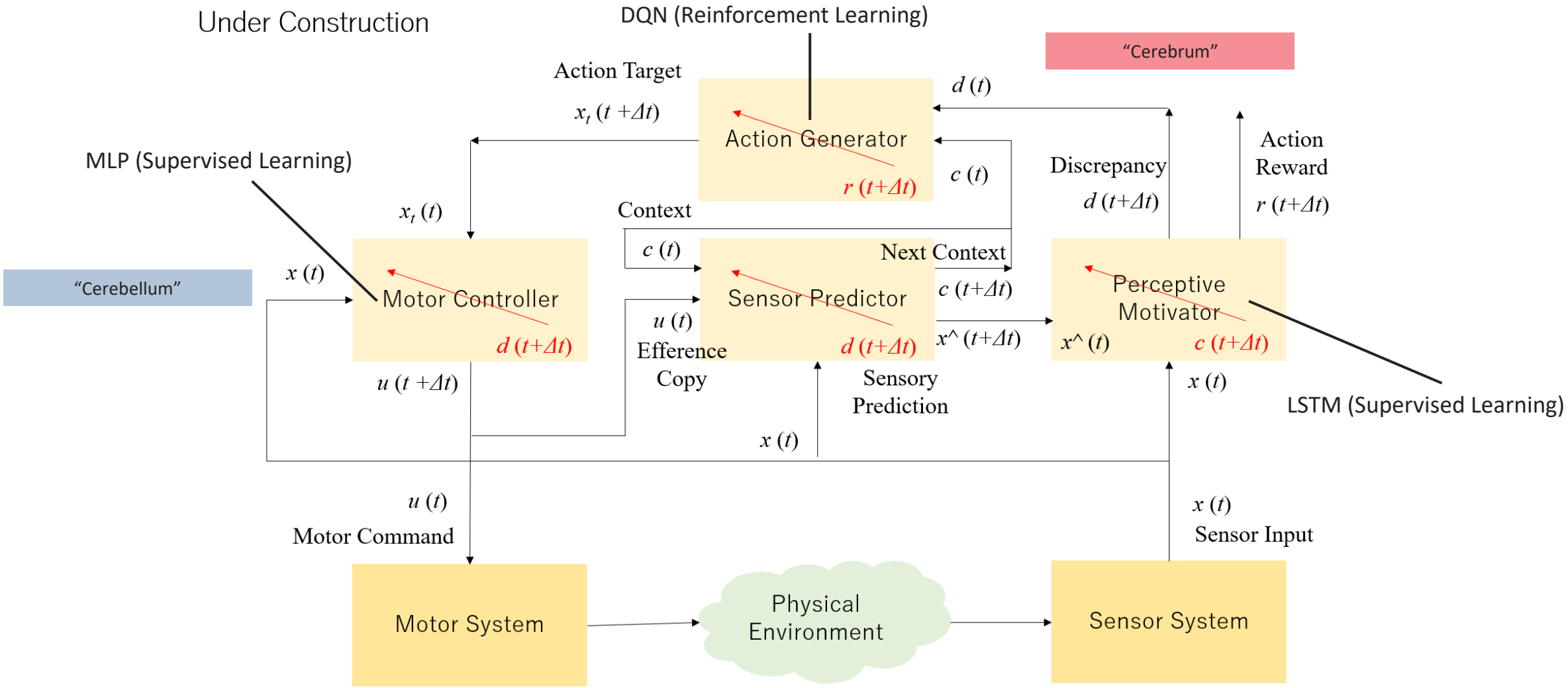




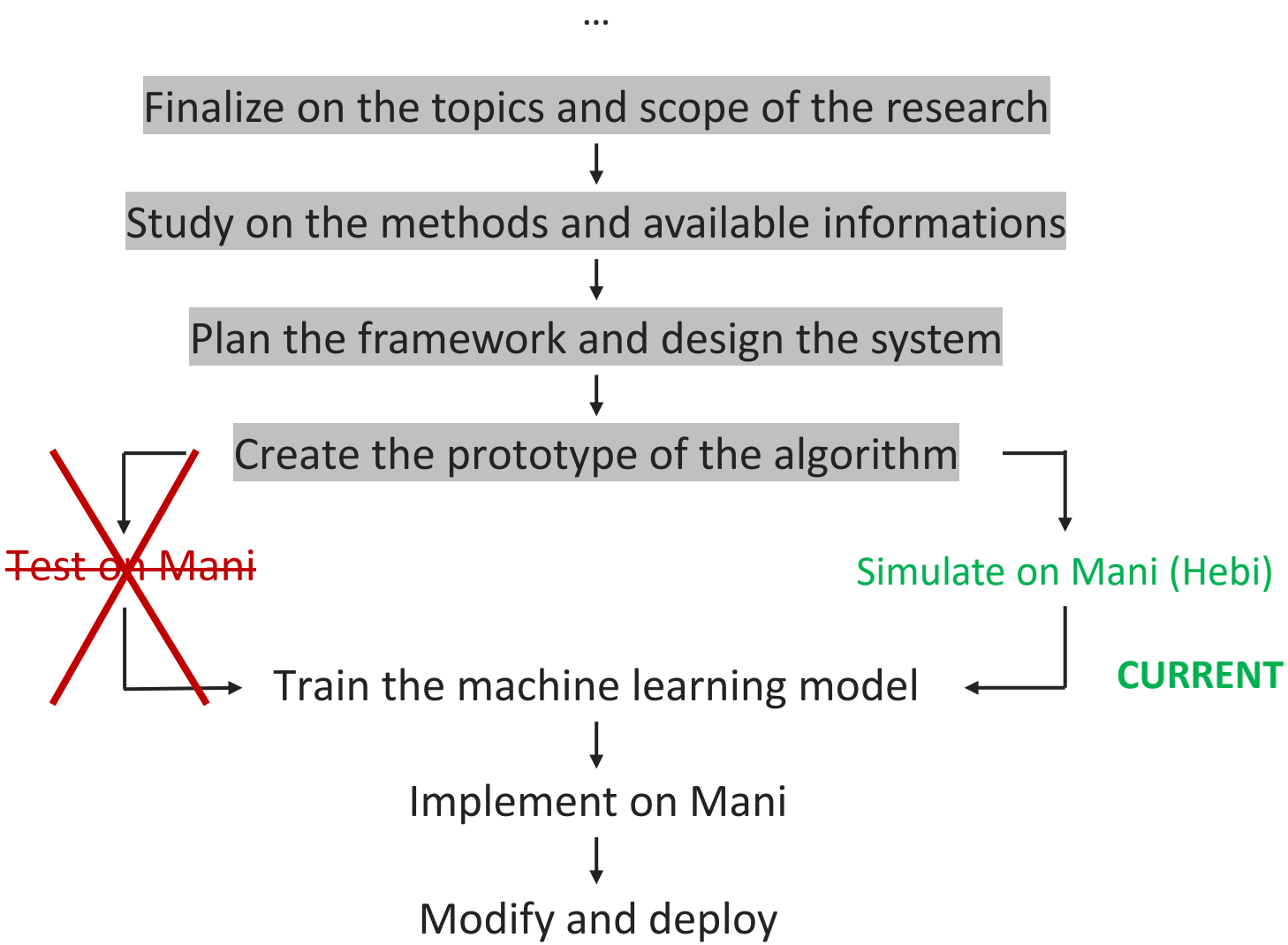
Marker Detection



The Algorithm Diagram



Progress


...
Finalize on the topics and scope of the research
Study on the methods and available informations
Plan the framework and design the system
Create the prototype of the algorithm
Test on Mani
Simulate on Mani (Hebi)
CURRENT
Train the machine learning model
Implement on Mani
Modify and deploy

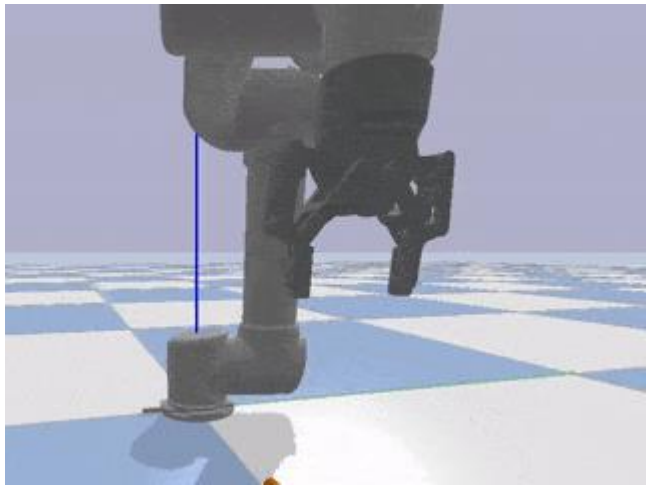
Methodology Refinement

- Primarily using simulations model instead of training with the real robot

PyTorch

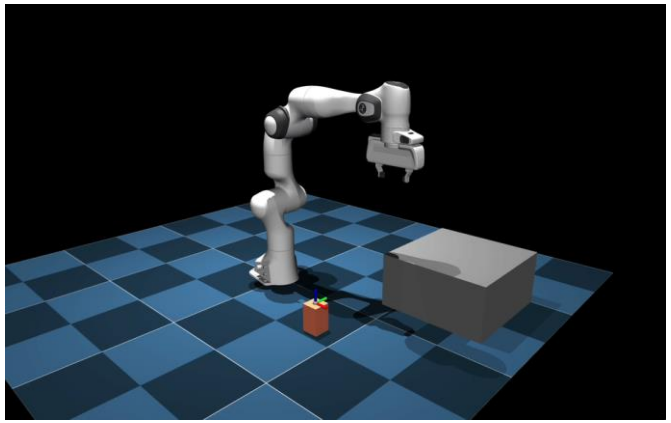
Gymnasium

[5]



PyBullet

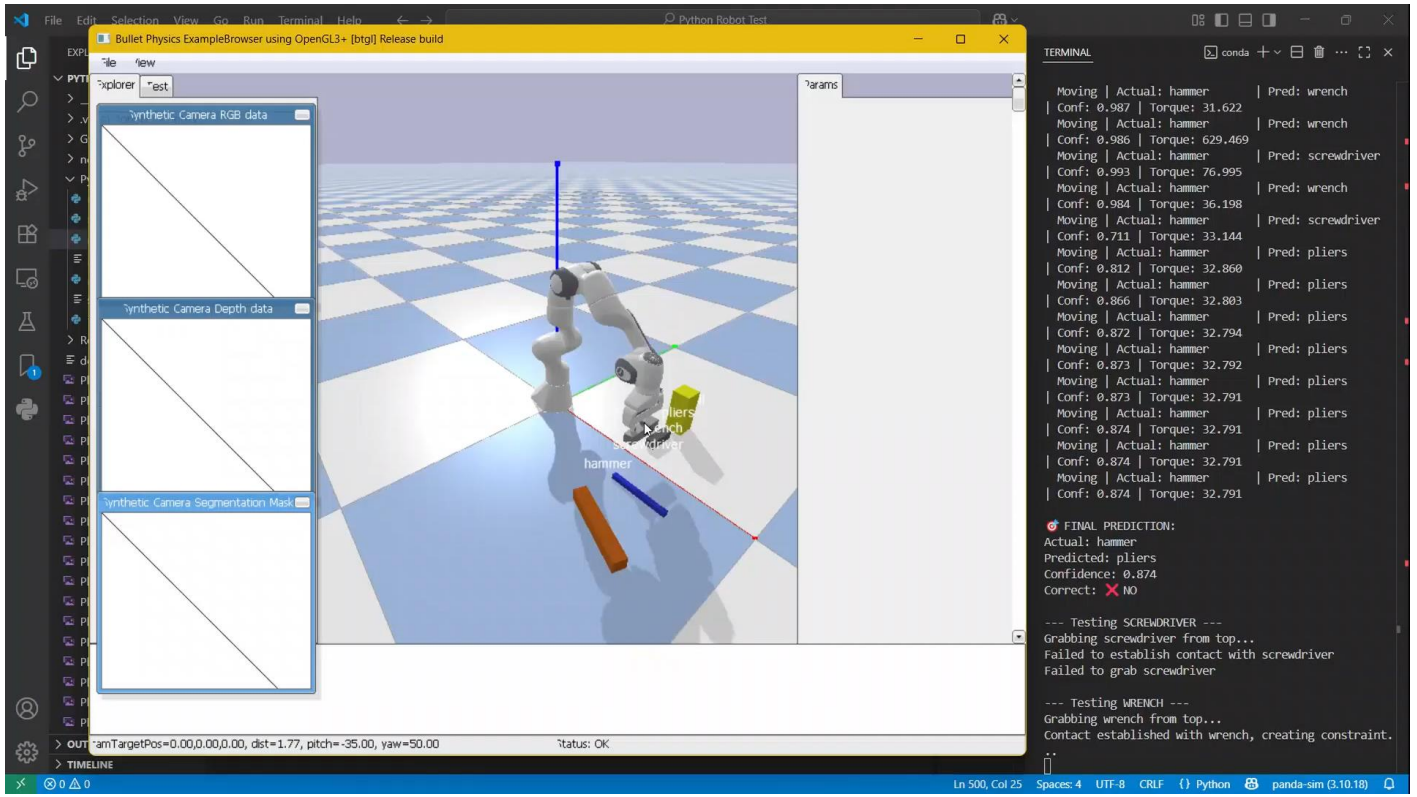
[6]



MuJoCo

Findings / Challenges (1)

- **<u>Tool Classification from The Saliency in Predicted Sensing.</u>**
- Using LSTM to predict the tools, mainly using torque sensor from the actuators.



Findings / Challenges (2)

- **A Goal Oriented Motor Planning for Tool Usage.**
- **Comparing DQN vs DLS**

DQN 🧠

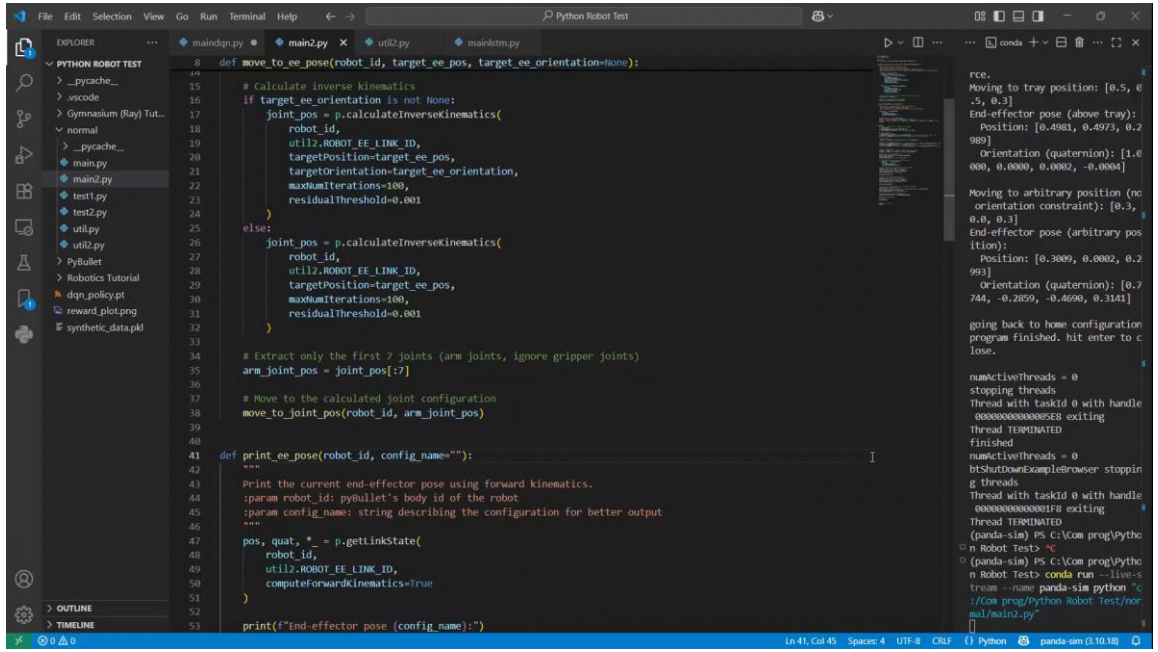
DQN is a **learning-based** approach. The robot learns through **trial and error**, using a **neural network** to decide actions that maximize future rewards. It observes its situation, acts, and gets feedback, continuously adjusting its network. This excels in complex, undefined tasks like **obstacle navigation** or **adaptive manipulation**.

DLS (Inverse Kinematics) ⚙

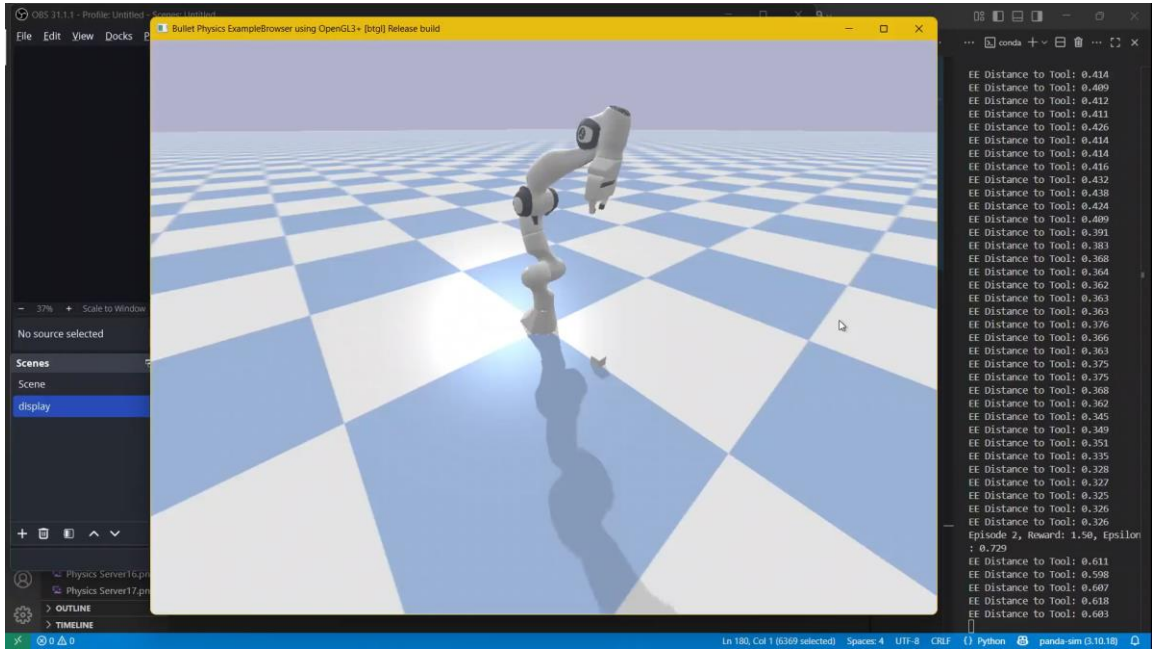
DLS is a **model-based mathematical technique**. It directly **calculates the specific joint angles** for a robot to reach a desired end-effector position, using the robot's **mathematical model**. DLS is ideal for tasks needing **high precision and repeatable movements**, like **industrial assembly**.

Findings / Challenges *(3)*

DLS



DQN



Findings / Challenges *(4)*

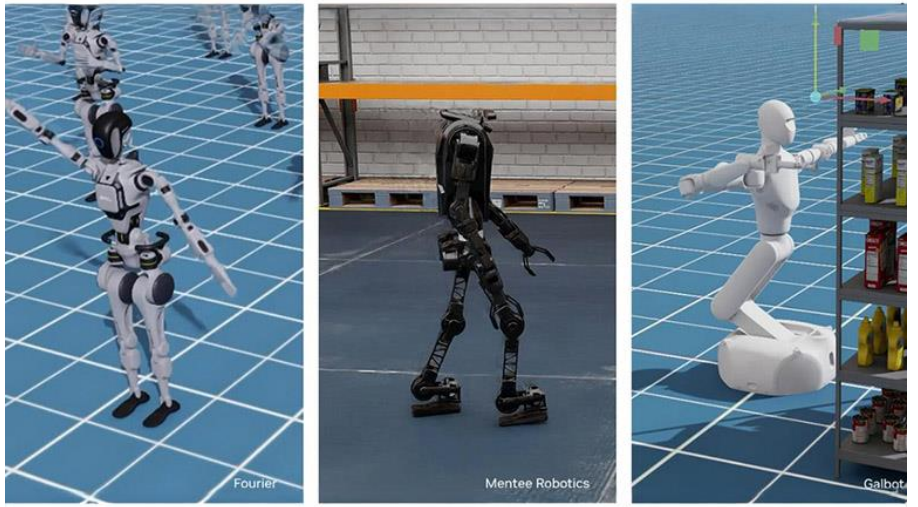
Main Challenges

- **ROS2 Environment Setup**: Setting up the software for ROS2 is often complex and causes many errors.
- **Robot Safety & Control**: Directly controlling a robot arm requires adding safety checks to every command to prevent problems.
- **DQN Training Time & Cost**: Training the DQN model takes a long time and use heavy hardware resources
- **Bridging Gap:** Models trained in simulation often don't work as well on a real robot due to differences in physics, sensors, and noise.

Remaining work

- **Bridging** connection between simulations and ROS2, to be able to train on the real robot model in the simulated environments. Increasing adaptability, accuracy from the machine learning model to the real-world robot.

[7]



Conclusions

From this project, I

- Learned about **Machine Learning Models** and **ROS2 Controls.**
- Built **machine learning algorithms** from scratch.
- Controlled a **real robot**, gaining hands-on experience.
- Improved my **programming skills** significantly.
- Built a **strong base** for future work in robot learning and control.