**BỘ CÔNG THƯƠNG**

**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP TP.HCM**

**KHOA CÔNG NGHỆ THÔNG TIN**

-----🙞🙜🕮🙞🙜-----



**KHÓA LUẬN TỐT NGHIỆP**

**ĐIỀU KHIỂN CÁNH TAY ROBOT SỬ DỤNG DEEP REINFORCEMENT LEARNING**

**GVHD: TS. ĐẶNG THỊ PHÚC**

|  |  |  |
| --- | --- | --- |
| Tên sinh viên | MSSV | Lớp |
| Lê Văn Tuấn Nguyên | **22642641** | **DHKHMT18CTT** |
| Trần Vũ Thùy Trang | **21129621** | **DHKHMT17ATT** |

TP.HCM, tháng 12 năm 2025

**MINISTRY OF INDUSTRY AND TRADE INDUSTRIAL UNIVERSITY OF HO CHI MINH CITY FACULTY OF INFORMATION TECHNOLOGY**

­­-----🙞🙜🕮🙞🙜-----



**GRADUATION THESIS**

**ROBOTIC ARM CONTROL USING DEEP REINFORCEMENT LEARNING**  
  
**Supervisor: Dr. Dang Thi Phuc**

|  |  |  |
| --- | --- | --- |
| Student name | Student code | Class |
| Le Van Tuan Nguyen | **22642641** | **DHKHMT18CTT** |
| Tran Vu Thuy Trang | **21129621** | **DHKHMT17ATT** |

HO CHI MINH CITY, December 2025

# ABSTRACT

This project presents a comprehensive comparative study of deep reinforcement learning algorithms for training robotic manipulators to perform complex object grasping and lifting tasks in simulated environments. The implementation combines a Franka FR3 robotic arm with a Shadow Hand end-effector, leveraging NVIDIA's Isaac Lab simulation framework to create a physically realistic training environment capable of handling diverse objects such as cubes, bottles, and tools through dynamic object selection mechanisms.

The research conducts extensive algorithmic comparisons. Soft Actor-Critic (SAC) with Relay Hindsight Experience Replay (RHER) as the primary approach, which employs Sequential GRU-based neural networks for both policy and value function approximation to capture temporal dependencies in multi-fingered grasping. For another algorithm comparison: Deep Deterministic Policy Gradient (DDPG) against SAC with both standard Hindsight Experience Replay (HER) and the relay variant (RHER), and Proximal Policy Optimization (PPO) as an on-policy alternative. This systematic comparison allows for rigorous evaluation of how different actor-critic architectures (deterministic vs. stochastic), experience replay mechanisms (HER vs. RHER), and policy optimization strategies (off-policy vs. on-policy) affect learning efficiency and final performance in high-dimensional manipulation tasks.

Custom reward shaping considers multiple factors including finger configurations to prevent self-collision, object proximity metrics based on normalized distance measures, and successful lifting criteria validated through both position and orientation thresholds. The trained policies exhibit reliable goal-conditioned manipulation across diverse objects, showcasing the potential of modern deep reinforcement learning techniques for solving complex robotic manipulation challenges in simulation and providing a foundation for future transfer to real-world robotic systems.

**Keywords**: Intelligent robotic arm control, 6D Pose Estimation, robotics, Deep Reinforcement Learning, Isaac Lab, Franka, Shadow Hand, SAC, RHER, SeqGRUNet, GPU.

# TÓM TẮT

Dự án này trình bày một nghiên cứu so sánh toàn diện về các thuật toán học tăng cường sâu (Deep Reinforcement Learning) để huấn luyện các tay máy robot thực hiện các nhiệm vụ gắp và nâng vật thể phức tạp trong môi trường mô phỏng. Việc triển khai kết hợp cánh tay robot Franka FR3 với bộ gắp Shadow Hand, tận dụng nền tảng mô phỏng Isaac Lab của NVIDIA để tạo ra một môi trường huấn luyện có tính thực tế vật lý, có khả năng xử lý nhiều loại vật thể như khối lập phương, chai lọ và dụng cụ thông qua cơ chế lựa chọn vật thể động.

Nghiên cứu tiến hành so sánh thuật toán một cách chuyên sâu. Soft Actor-Critic (SAC) kết hợp với Relay Hindsight Experience Replay (RHER) được sử dụng làm phương pháp chính, áp dụng mạng nơ-ron GRU tuần tự cho cả việc xấp xỉ hàm chính sách và hàm giá trị nhằm nắm bắt sự phụ thuộc theo thời gian trong thao tác gắp nhiều ngón. Ngoài ra, nghiên cứu còn so sánh các thuật toán khác: Deep Deterministic Policy Gradient (DDPG) với SAC (kết hợp HER chuẩn và biến thể RHER), cùng với Proximal Policy Optimization (PPO) như một lựa chọn theo chính sách (on-policy). Sự so sánh có hệ thống này cho phép đánh giá chặt chẽ cách các kiến trúc actor-critic khác nhau (xác định vs. ngẫu nhiên), cơ chế phát lại kinh nghiệm (HER vs. RHER) và chiến lược tối ưu hóa chính sách (off-policy vs. on-policy) ảnh hưởng đến hiệu quả học tập và hiệu suất cuối cùng trong các nhiệm vụ thao tác có không gian trạng thái cao chiều.

Việc thiết kế hàm thưởng tùy chỉnh xem xét nhiều yếu tố, bao gồm cấu hình ngón tay để tránh va chạm, các chỉ số khoảng cách chuẩn hóa để đánh giá độ gần của vật thể, và tiêu chí nâng thành công được xác thực thông qua cả ngưỡng vị trí và hướng. Các chính sách được huấn luyện thể hiện khả năng thao tác theo mục tiêu một cách đáng tin cậy trên nhiều loại vật thể, chứng minh tiềm năng của các kỹ thuật học tăng cường sâu hiện đại trong việc giải quyết các thách thức thao tác robot phức tạp trong mô phỏng và tạo nền tảng cho việc chuyển giao sang hệ thống robot thực tế.

**Keywords**: Intelligent robotic arm control, 6D Pose Estimation, robotics, Deep Reinforcement Learning, Isaac Lab, Franka, Shadow Hand, SAC, RHER, SeqGRUNet, GPU, tự động hóa.

# LỜI CAM ĐOAN

Em xin cam đoan việc nghiên cứu đề tài được thực hiện cùng với giáo viên hướng dẫn khóa luận và chưa được sử dụng để báo cáo cho bất kì đồ án hay môn học nào.

Em xin cam đoan mọi sự giúp đỡ đã được cảm ơn, các thông tin trích dẫn đã được ghi chú đầy đủ và bảo đảm rõ nguồn gốc không xâm phạm quyền tác giả.

TP. Hồ Chí Minh, tháng 12 năm 2025

TÁC GIẢ KHÓA LUẬN

Trần Vũ Thùy Trang

Lê Văn Tuấn Nguyên

# LỜI CẢM ƠN

Quá trình làm khóa luận tốt nghiệp là giai đoạn quan trọng nhất trong quãng đời mỗi sinh viên. Khóa luận tốt nghiệp là tiền đề, nhằm trang bị cho chúng em những kỹ năng nghiên cứu, những kiến thức quý báu trước khi lập nghiệp.

Trước hết, chúng em xin cảm ơn quý thầy cô Khoa Công nghệ thông tin. Đặc biệt là thầy cô trong bộ môn Khoa học Máy tính đã tận tình chỉ dạy và trang bị cho chúng em những kiến thức cần thiết trong suốt thời gian ngồi trên ghế giảng đường. Làm nền tảng cho chúng em hoàn thành được khóa luận này.

Đặc biệt, chúng em xin cảm ơn giảng viên TS. Đặng Thị Phúc, cô đã tận tình hướng dẫn và hỗ trợ trong suốt 5 năm học đại học. Đó là những sự hỗ trợ hết sức quý báu trong quá trình thực hiện khóa luận này và trong tương lai.

Và cuối cùng, xin gửi lời cảm ơn đến gia đình, bạn bè, những người luôn sẵn sàng chia sẻ và giúp đỡ trong học tập và cuộc sống. Mong rằng chúng ta đều sẽ vững bước trong tương lai.

Với khả năng của bản thân và kiến thức chưa đủ sâu sắc sẽ không thể tránh khỏi việc thiếu sót và hạn chế, kính mong sự đóng góp của cô để khóa luận của nhóm được hoàn thiện hơn.

Em xin trân trọng cảm ơn!

TP. Hồ Chí Minh, tháng 12 năm 2025

**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

......................................................................................................................................

......................................................................................................................................

.....................................................................................................................................

......................................................................................................................................

......................................................................................................................................

......................................................................................................................................

......................................................................................................................................

......................................................................................................................................

......................................................................................................................................

......................................................................................................................................

......................................................................................................................................

......................................................................................................................................

......................................................................................................................................

......................................................................................................................................

......................................................................................................................................

......................................................................................................................................

TP. Hồ Chí Minh, tháng 12 năm 2025

GIẢNG VIÊN HƯỚNG DẪN

TS. ĐẶNG THỊ PHÚC

**NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN**

......................................................................................................................................

......................................................................................................................................

......................................................................................................................................

......................................................................................................................................

......................................................................................................................................

......................................................................................................................................

......................................................................................................................................

......................................................................................................................................

......................................................................................................................................

......................................................................................................................................

......................................................................................................................................

......................................................................................................................................

......................................................................................................................................

......................................................................................................................................

......................................................................................................................................

......................................................................................................................................

TP. Hồ Chí Minh, tháng 12 năm 2025

GIẢNG VIÊN PHẢN BIỆN

# MỤC LỤC

[CHƯƠNG 1. TỔNG QUAN ĐỀ TÀI 17](#_Toc215569545)

[1.1. Lý do chọn đề tài 17](#_Toc215569546)

[1.2. Mục tiêu và nhiệm vụ nghiên cứu 18](#_Toc215569547)

[1.2.1. Mục tiêu nghiên cứu 18](#_Toc215569548)

[1.2.2. Nhiệm vụ nghiên cứu 19](#_Toc215569549)

[1.3. Đối tượng nghiên cứu 23](#_Toc215569550)

[1.3.1. Hệ thống robot 23](#_Toc215569551)

[1.3.2. Thuật toán điều khiển 23](#_Toc215569552)

[1.3.3. Môi trường mô phỏng 23](#_Toc215569553)

[1.4. Phạm vi nghiên cứu 23](#_Toc215569554)

[1.5. Phương pháp nghiên cứu 24](#_Toc215569555)

[1.5.1. Phương pháp Nghiên cứu Lý thuyết 24](#_Toc215569556)

[1.5.2. Phương pháp Mô phỏng (Simulation-based Approach) 25](#_Toc215569557)

[1.5.3. Phương pháp Thực nghiệm So sánh (Comparative Experimental Method) 25](#_Toc215569558)

[1.5.4. Phương pháp Thiết kế Hàm thưởng (Reward Shaping) 26](#_Toc215569559)

[1.5.5. Phương pháp Đánh giá Định lượng 26](#_Toc215569560)

[1.6. Ý nghĩa khoa học và thực tiễn 27](#_Toc215569561)

[CHƯƠNG 2. CƠ SỞ LÝ THUYẾT 28](#_Toc215569562)

[2.1. Tổng quan 28](#_Toc215569563)

[2.1.1. Hệ thống điều khiển cánh tay robot truyền thống 28](#_Toc215569564)

[2.1.2. Bối cảnh và động lực nghiên cứu 29](#_Toc215569565)

[2.1.3. Phương pháp tiếp cận của nghiên cứu 31](#_Toc215569566)

[2.2. Cơ sở Robot học (Robotics) 33](#_Toc215569567)

[2.2.1. Động học robot (Kinematics) 33](#_Toc215569568)

[2.2.1.1. Động học thuận (Forward Kinematics) 33](#_Toc215569569)

[2.2.1.2. Động học ngược (Inverse Kinematics - IK) 35](#_Toc215569570)

[2.2.2. Động lực học robot (Robot Dynamics) 36](#_Toc215569571)

[2.2.3. Lý thuyết cầm nắm (Grasping Theory) 37](#_Toc215569572)

[2.2.3.1. Form Closure 37](#_Toc215569573)

[2.2.3.2. Force Closure 38](#_Toc215569574)

[2.2.3.3. Chất lượng của grasp (Grasp Quality Metrics) 38](#_Toc215569575)

[2.2.4. Mô hình hóa robot trong mô phỏng 39](#_Toc215569576)

[2.2.4.1. Khái niệm link và joint trong robot 39](#_Toc215569577)

[2.2.4.2. Mô hình hóa robot trong mô phỏng Isaacsim 39](#_Toc215569578)

[2.2.4.3. Quy trình đưa robot từ đời thực vào mô phỏng 40](#_Toc215569579)

[2.2.5. Cấu tạo hệ thống robot sử dụng 42](#_Toc215569580)

[2.3. Học tăng cường (Reinforcement Learning) 46](#_Toc215569581)

[2.3.1. Giới thiệu và khái niệm cơ bản 46](#_Toc215569582)

[2.3.2. Các thành phần cốt lõi 47](#_Toc215569583)

[2.3.3. Quy trình ra quyết định Markov (Markov Decision Process - MDP) 52](#_Toc215569584)

[2.4. Học Sâu (Deep Learning) và Học Tăng Cường Sâu (DRL 58](#_Toc215569585)

[2.4.1. Từ RL đến DRL: Vai trò của Deep Neural Networks 58](#_Toc215569586)

[2.4.2. Mạng nơ-ron trong DRL 60](#_Toc215569587)

[2.4.2.1. Value Function Approximation 60](#_Toc215569588)

[2.4.2.2. Policy Function Approximation 60](#_Toc215569589)

[2.4.3. Tổng quan các thuật toán DRL cho điều khiển robot liên tục 62](#_Toc215569590)

[2.4.3.1. PPO (Proximal Policy Optimization) 64](#_Toc215569591)

[2.4.3.2. DDPG (Deep Deterministic Policy Gradient) 65](#_Toc215569592)

[2.4.4. Các thuật toán Actor-Critic 67](#_Toc215569593)

[2.4.4.1. Kiến trúc Actor-Critic 67](#_Toc215569594)

[2.4.4.2. Actor (Policy Network) 68](#_Toc215569595)

[2.4.4.3. Critic (Value Network) 69](#_Toc215569596)

[2.5. Goal-Conditioned Reinforcement Learning (GCRL) 69](#_Toc215569597)

[2.5.1. Động lực và khái niệm 69](#_Toc215569598)

[2.5.2. Formulation của GCRL 70](#_Toc215569599)

[2.5.3. Ưu điểm trong bài toán đa nhiệm vụ và đa mục tiêu 73](#_Toc215569600)

[2.5.4. Áp dụng trong nghiên cứu này 73](#_Toc215569601)

[2.6. Thuật toán Soft Actor-Critic (SAC) 74](#_Toc215569602)

[2.6.1. Maximum Entropy Reinforcement Learning 74](#_Toc215569603)

[2.6.2. Entropy Regularization 75](#_Toc215569604)

[2.6.3. Công thức toán học của SAC 76](#_Toc215569605)

[2.6.4. Ưu điểm của SAC 78](#_Toc215569606)

[2.6.5. Nhược điểm 79](#_Toc215569607)

[2.7. Tích hợp SAC với GCRL (SAC-GCRL) 80](#_Toc215569608)

[2.7.1. Kiến trúc tổng thể 80](#_Toc215569609)

[2.7.2. Multi-stage Goal Structure 81](#_Toc215569610)

[2.7.3. Entropy-regularized optimization cho từng stage 83](#_Toc215569611)

[2.7.4. Training Procedure 84](#_Toc215569612)

[2.8. Hindsight Experience Replay (HER) 86](#_Toc215569613)

[2.8.1. Vấn đề sparse reward trong robot learning 86](#_Toc215569614)

[2.8.2. Cơ chế Goal Relabeling 87](#_Toc215569615)

[2.8.3. Các chiến lược sampling 87](#_Toc215569616)

[2.8.4. Công thức HER Relabeling 88](#_Toc215569617)

[2.9. Relay Hindsight Experience Replay (RHER) 90](#_Toc215569618)

[2.9.1. Động lực và thách thức 90](#_Toc215569619)

[2.9.2. Cơ chế Hindsight Experience Replay (HER) 90](#_Toc215569620)

[2.9.3. Kiến trúc RHER 92](#_Toc215569621)

[2.9.4. Thuật toán RHER 94](#_Toc215569622)

[2.9.5. Kỹ thuật huấn luyện 95](#_Toc215569623)

[2.9.5.1. N-step Returns 95](#_Toc215569624)

[2.9.5.1. Recurrent Batching 95](#_Toc215569625)

[2.9.5.2. Chiến lược Priority và Sampling 95](#_Toc215569626)

[2.10. Kiến trúc mạng nơ-ron (Neural Network Architecture) 96](#_Toc215569627)

[2.10.1. Policy Network (SeqGRUNet) 96](#_Toc215569628)

[2.10.2. Value Network (Critic) 98](#_Toc215569629)

[2.10.3. Target Networks 99](#_Toc215569630)

[2.10.4. Temperature và Penalty Coefficients 101](#_Toc215569631)

[2.11. Tổng quan hệ thống nghiên cứu 102](#_Toc215569632)

[CHƯƠNG 3. KẾT QUẢ THỰC NGHIỆM 105](#_Toc215569633)

[3.1. Cấu hình môi trường 105](#_Toc215569634)

[3.1.1. Tổng quan về Isaac Sim 105](#_Toc215569635)

[3.1.2. Cấu hình môi trường mô phỏng 105](#_Toc215569636)

[3.1.3. Các đối tượng được mô phỏng 105](#_Toc215569637)

[3.1.4. Mô phỏng tiếp xúc và vật lý 109](#_Toc215569638)

[3.1.5. Tích hợp mô phỏng với hệ thống học tăng cường sâu 110](#_Toc215569639)

[3.1.6. So sánh với các nền tảng mô phỏng khác 111](#_Toc215569640)

[3.2. Học tăng cường sâu (DRL) 113](#_Toc215569641)

[3.2.1. Môi trường thực nghiệm 113](#_Toc215569642)

[3.2.2. Không gian trạng thái (Observation Space) 113](#_Toc215569643)

[3.2.3. Không gian hành động (Action Space) 115](#_Toc215569644)

[3.2.4. Multi-stage Goal Encoding 116](#_Toc215569645)

[3.2.5. Hàm phần thưởng (Reward Function) 119](#_Toc215569646)

[3.2.6. Điều kiện kết thúc Episode (Termination) 119](#_Toc215569647)

[3.2.7. Domain Randomization 119](#_Toc215569648)

[3.2.7.1. Object Randomization 119](#_Toc215569649)

[3.2.7.2. Initial Pose Randomization 121](#_Toc215569650)

[3.2.7.3. Goal Position Randomization 122](#_Toc215569651)

[3.3. Quy trình huấn luyện 122](#_Toc215569652)

[CHƯƠNG 4. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 132](#_Toc215569653)

# MỤC LỤC BẢNG BIỂU

[Bảng 2.1: Đặc tính các khớp của robot Franka Emika FR3 43](#_Toc215569830)

[Bảng 2.2: Đặc tính các khớp của bàn tay Shadow Hand 44](#_Toc215569831)

[Bảng 2.2: Các thành phần kiến trúc mô hình 97](#_Toc215569832)

[Bảng 3.1: Các thành phần cố định của môi trường 106](#_Toc215569833)

[Bảng 3.2: Các vật thể dùng cho thao tác 107](#_Toc215569834)

[Bảng 2.3: So sánh Isaacsim, Mujoco và Pybullet 111](#_Toc215569835)

[Bảng 3.1: Định nghĩa các thành phần trong mỗi trạng thái 113](#_Toc215569836)

[Bảng 3.3: Thông số vật lý của các vật thể 120](#_Toc215569837)

[Bảng 3.4: So sánh Object Initial Position và Goal Ranges 122](#_Toc215569838)

[Bảng 3.5: So sánh đặc điểm và hiệu quả của các thuật toán 124](#_Toc215569839)

[Bảng 3.6: So sánh hiệu năng của các phương pháp trên tác vụ gắp 129](#_Toc215569840)

# MỤC LỤC HÌNH ẢNH

[Hình 2.1: Sơ đồ quy trình điều khiển cánh tay Robot 25](#_Toc215508790)

[Hình 2.2: Sự giao thoa giữa ba lĩnh vực tạo nên DRL 28](#_Toc215508791)

[Hình 2.3: Sự giao thoa giữa ba lĩnh vực tạo nên DRL 29](#_Toc215508792)

[Hình 2.4: Cấu hình khớp của cánh tay Franka Emika FR3 31](#_Toc215508793)

[Hình 2.5: Sơ đồ về cách thức hoạt động của học tăng cường (RL) 46](#_Toc215508794)

[Hình 2.6: Phân loại thuật toán DRL 55](#_Toc215508795)

[Hình 2.7: Luồng xử lý PPO/SAC 59](#_Toc215508796)

[Hình 2.8: Actor Critic Process 60](#_Toc215508797)

[Hình 2.9: SAC-GCRL Agent Architecture 71](#_Toc215508798)

[Hình 2.10: Multi-stage Goal Structure 72](#_Toc215508799)

[Hình 2.12: Phân rã và sắp xếp lại các tác vụ theo trình tự. 84](#_Toc215508800)

[Hình 2.13: Tổng quan kiến trúc mô hình 91](#_Toc215508801)

[Hình 2.14: Thiết lập môi trường thực nghiệm 95](#_Toc215508802)

[Hình 2.15: Cánh tay robot thực hiện tác vụ 95](#_Toc215508803)

[Hình 2.16: Cánh tay robot hoàn thành tác vụ 96](#_Toc215508804)

[Hình 3.1: Kiến trúc các mạng neural networks được sử dụng cho các thuật toán DRL 108](#_Toc215508805)

[Hình 3.2: Hiệu năng của các phương pháp trên tác vụ gắp ( Distance ) 109](#_Toc215508806)

[Hình 3.3: Hiệu năng của các phương pháp trên tác vụ gắp ( Goal\_achived ) 109](#_Toc215508807)

# MỤC LỤC TỪ VIẾT TẮT

|  |  |  |
| --- | --- | --- |
| **Từ viết tắt** | **Từ đầy đủ** | **Nghĩa tiếng Việt** |
| AI | Artificial Intelligence | Trí tuệ nhân tạo |
| CNN | Convolutional Neural Network | Mạng nơ-ron tích chập |
| RL | Reinforcement Learning | Học tăng cường |
| DRL | Deep Reinforcement Learning | Học tăng cường sâu |
| MDP | Markov Decision Process | Quy trình quyết định Markov |
| HER | Hindsight Experience Replay | Phát lại kinh nghiệm hồi tưởng |
| RHER | Relay Hindsight Experience Replay | Phát lại kinh nghiệm hồi tưởng dạng tiếp sức |
| SAC | Soft Actor-Critic | Thuật toán Soft Actor-Critic |
| DDPG | Deep Deterministic Policy Gradient | Thuật toán Deep Deterministic Policy Gradient |
| PPO | Proximal Policy Optimization | Thuật toán Proximal Policy Optimization |
| TD3 | Twin Delayed DDPG | Thuật toán Twin Delayed DDPG |
| GCRL | Goal-Conditioned Reinforcement Learning | Học tăng cường có điều kiện mục tiêu |
| GRU | Gated Recurrent Unit | Đơn vị hồi tiếp có cổng |
| SeqGRUNet | Sequential GRU Network | Mạng GRU tuần tự |
| GPU | Graphics Processing Unit | Bộ xử lý đồ họa |
| ADD | Average Distance of Model Points | Khoảng cách trung bình của các điểm mô hình |
| ADD-S | Average closest point distance | Khoảng cách điểm gần nhất trung bình |
| DoF | Degrees of Freedom | Số bậc tự do |
| MTL | Multi-task Learning | Học đa nhiệm vụ |
| SGD | Stochastic Gradient Descent | Thuật toán tối ưu hóa ngẫu nhiên |
| MLP | Multi-Layer Perceptron | Mạng nơ-ron nhiều lớp |
| LSTM | Long Short-Term Memory | Bộ nhớ ngắn dài hạn |
| EMA | Exponential Moving Average | Trung bình động hàm mũ |
| API | Application Programming Interface | Giao diện lập trình ứng dụng |
| USD | Universal Scene Description | Định dạng mô tả cảnh phổ quát (NVIDIA) |
| RTX | Ray Tracing | Công nghệ dò tia (NVIDIA) |
| AdamW | Adaptive Moment Estimation with Weight Decay | Bộ tối ưu hóa Adam có suy giảm trọng số |
| Sim-to-Real | Simulation to Real | Chuyển giao từ mô phỏng sang thực tế |
| POMDP | Partially Observable Markov Decision Process | Quy trình quyết định Markov quan sát một phần |
| Q-value | Quality Value | Giá trị chất lượng (trong RL) |
| MSE | Mean Squared Error | Sai số bình phương trung bình |
| GWS | Grasp Wrench Space | Không gian lực nắm |
| MCP | Metacarpophalangeal | Khớp bàn tay (giải phẫu) |
| PIP | Proximal Interphalangeal | Khớp gần ngón tay (giải phẫu) |
| DIP | Distal Interphalangeal | Khớp xa ngón tay (giải phẫu) |
| Adam | Adaptive Moment Estimation | Bộ tối ưu hóa Adam |
| GAE | Generalized Advantage Estimation | Ước lượng lợi thế tổng quát |

1. TỔNG QUAN ĐỀ TÀI
   1. Lý do chọn đề tài

Trong bối cảnh tự động hóa và robot công nghiệp ngày càng phát triển, việc trang bị cho robot khả năng thao tác với các vật thể đa dạng trong môi trường phi cấu trúc vẫn là một trong những thách thức cốt lõi và cấp thiết nhất của ngành robot học. Các tác vụ như tiếp cận (reaching), cầm nắm (grasping), và nâng vật thể (lifting), vốn đơn giản đối với con người, lại đòi hỏi sự phối hợp cực kỳ phức tạp về điều khiển, tri giác và lập kế hoạch ở robot, đặc biệt là với các hệ thống có bậc tự do cao như cánh tay robot Franka FR3 (7-DoF) kết hợp bàn tay khéo léo Shadow Dexterous Hand (24-DoF), tổng cộng 31 bậc tự do. Thao tác cầm nắm khéo léo không chỉ là nền tảng cho các ứng dụng sản xuất linh hoạt mà còn mở ra tiềm năng cho robot trong các lĩnh vực dịch vụ, y tế và hậu cần.

Sự trỗi dậy của các phương pháp học tăng cường sâu (Deep Reinforcement Learning - DRL) trong những năm gần đây đã mang lại một hướng đi đầy hứa hẹn để giải quyết bài toán này. Thay vì lập trình thủ công các quy tắc điều khiển phức tạp, DRL cho phép robot tự học các chiến lược tối ưu thông qua quá trình tương tác và thử-sai trực tiếp với môi trường. Tuy nhiên, một câu hỏi lớn vẫn còn bỏ ngỏ trong cộng đồng nghiên cứu: đâu là kiến trúc thuật toán, cơ chế học hỏi và chiến lược tối ưu hóa hiệu quả nhất cho bài toán thao tác đa giai đoạn (multi-stage manipulation) với không gian hành động liên tục và phần thưởng thưa thớt (sparse rewards)

Hiện có nhiều phương pháp DRL tiên tiến như Soft Actor-Critic (SAC), Deep Deterministic Policy Gradient (DDPG), và Proximal Policy Optimization (PPO), cùng với các kỹ thuật bổ trợ như Hindsight Experience Replay (HER) và Relay Hindsight Experience Replay (RHER) để xử lý phần thưởng thưa thớt. Tuy nhiên, việc lựa chọn và đánh giá chúng một cách có hệ thống trong cùng một điều kiện tiêu chuẩn, đặc biệt là trên các hệ thống robot có cấu trúc phức tạp với bàn tay 5 ngón khéo léo (dexterous hand), vẫn chưa được thực hiện đầy đủ.

Xuất phát từ những thách thức và tiềm năng đó, đề tài "Huấn luyện cánh tay robot sử dụng học tăng cường sâu" được lựa chọn với trọng tâm là hệ thống Franka FR3 + Shadow Dexterous Hand thực hiện tác vụ nâng vật thể (object lifting) trong môi trường mô phỏng NVIDIA Isaac Lab. Mục tiêu chính của nghiên cứu này không chỉ là xây dựng một chính sách điều khiển thành công cho tác vụ reach-grasp-lift với 31 bậc tự do, mà còn thực hiện một cuộc khảo sát so sánh toàn diện và sâu rộng giữa các phương pháp học tăng cường tiên tiến.

Bằng cách đối chiếu một cách có hệ thống hiệu suất giữa các thuật toán tiêu biểu (SAC, DDPG, PPO) kết hợp với các cơ chế học hỏi kinh nghiệm (HER, RHER), và đặc biệt là kiến trúc mạng hồi tiếp SeqGRUNet để xử lý chuỗi quan sát theo thời gian, nghiên cứu hướng đến việc cung cấp những hiểu biết khoa học giá trị về ưu và nhược điểm của từng phương pháp trong bối cảnh goal-conditioned reinforcement learning đa giai đoạn. Kết quả của nghiên cứu sẽ đóng góp vào cơ sở lý thuyết và thực tiễn, làm tiền đề cho việc lựa chọn và phát triển các giải pháp học tăng cường hiệu quả hơn, không chỉ trong mô phỏng mà còn định hướng cho việc chuyển giao sang các hệ thống robot vật lý (sim-to-real transfer) trong tương lai.

* 1. Mục tiêu và nhiệm vụ nghiên cứu
     1. Mục tiêu nghiên cứu

Nghiên cứu và phát triển hệ thống điều khiển robot tự động dựa trên học tăng cường sâu (Deep Reinforcement Learning - DRL) để thực hiện tác vụ thao tác khéo léo (dexterous manipulation), cụ thể là cầm, nắm và nhấc vật thể (grasping and lifting) với khả năng tổng quát hóa cao trên nhiều loại vật thể khác nhau.

Mục tiêu cụ thể

Mục tiêu 1 - Điều khiển đa ngón phối hợp:

Phát triển chính sách điều khiển cho phép robot điều khiển đồng bộ và phối hợp nhiều ngón của bàn tay Shadow Hand

* Tạo ra các thế nắm (grasp configuration) ổn định và thích ứng với hình dạng, kích thước vật thể khác nhau
* Đảm bảo sự linh hoạt trong việc điều chỉnh lực và vị trí từng ngón tay

**Mục tiêu 2 - Khả năng tổng quát hóa với vật thể đa dạng**:

* Xây dựng chính sách điều khiển đủ robust để xử lý nhiều loại vật thể khác nhau (khối lập phương, chai, dụng cụ, v.v.)
* Không yêu cầu tinh chỉnh hoặc huấn luyện lại cho từng loại vật thể cụ thể
* Thích ứng với vật thể xuất hiện tại các vị trí và hướng ngẫu nhiên trong không gian làm việc

**Mục tiêu 3 - Hoàn thành tác vụ thao tác phức hợp**:

* Robot không chỉ nắm được vật thể mà còn phải nhấc nó lên khỏi mặt bàn
* Duy trì vật thể ở độ cao mục tiêu nhất định trong một khoảng thời gian
* Kiểm soát và duy trì hướng (orientation) của vật thể trong quá trình thao tác
* Đảm bảo độ ổn định và an toàn trong suốt quá trình thực hiện tác vụ

**Mục tiêu 4 - Tối ưu hóa hiệu suất và hiệu quả**:

* Đạt được tỷ lệ thành công cao (success rate > 80%) trong môi trường mô phỏng
* Tối thiểu hóa thời gian huấn luyện và số lượng mẫu tương tác cần thiết
* Đảm bảo khả năng điều khiển thời gian thực với độ trễ thấp.
  + 1. Nhiệm vụ nghiên cứu

Nghiên cứu nền tảng lý thuyết:

* Tìm hiểu sâu về các khái niệm cốt lõi của học tăng cường (Reinforcement Learning) và học tăng cường sâu (Deep Reinforcement Learning)
* Nghiên cứu các kiến trúc mạng neural phù hợp (Actor-Critic, Policy Gradient, Value-based methods)
* Phân tích các công trình nghiên cứu tiên phong trong lĩnh vực robot manipulation và dexterous grasping

Khảo sát thuật toán DRL:

* So sánh các họ thuật toán: PPO (Proximal Policy Optimization), SAC (Soft Actor-Critic), TD3 (Twin Delayed DDPG), DDPG, A3C
* Phân tích ưu nhược điểm của từng thuật toán trong bối cảnh điều khiển robot
* Lựa chọn thuật toán phù hợp với yêu cầu bài toán

Xây dựng môi trường mô phỏng với NVIDIA Isaac Lab:

* Thiết lập môi trường mô phỏng vật lý chân thực với động lực học chính xác
* Mô hình hóa cánh tay robot Franka Emika FR3 (7 bậc tự do)
* Tích hợp bàn tay Shadow Hand với đầy đủ các khớp và actuators
* Cấu hình các thông số vật lý: ma sát, độ cứng, khối lượng, moment quán tính

Thiết kế không gian vật thể và tương tác:

* Lựa chọn và tạo tập dữ liệu vật thể đa dạng (geometric shapes, household objects, tools)
* Thiết lập cơ chế sinh vật thể ngẫu nhiên (position, orientation, type)
* Xây dựng cơ chế phát hiện va chạm và tương tác vật lý giữa robot-vật thể

Định nghĩa không gian trạng thái và hành động:

* Xác định observation space: vị trí/góc khớp, vận tốc, lực tiếp xúc, vị trí/hướng vật thể
* Xác định action space: điều khiển từng khớp của cánh tay và bàn tay
* Thiết lập giới hạn và ràng buộc an toàn

Phát triển reward shaping hiệu quả:

* Thiết kế hàm phần thưởng đơn giản nhưng hiệu quả, không phụ thuộc kiến thức chuyên gia
* Kết hợp nhiều thành phần reward: distance reward, grasp reward, lift reward, orientation reward
* Cân bằng giữa sparse reward và dense reward để tăng tốc độ học

Tối ưu hóa reward structure:

* Thiết lập các milestone rewards cho các giai đoạn: tiếp cận → nắm → nhấc → giữ
* Xử lý negative rewards cho các hành vi không mong muốn (rơi vật thể, va chạm)
* Đảm bảo tính tổng quát, không cần điều chỉnh cho từng loại vật thể

Hiện thực hóa thuật toán:

* Cài đặt các thuật toán DRL đã chọn (ví dụ: PPO, SAC) bằng Python
* Sử dụng các framework hỗ trợ: PyTorch, Stable-Baselines3, hoặc tự implement
* Tích hợp với môi trường Isaac Lab thông qua Gym interface

Thiết lập pipeline huấn luyện:

* Xây dựng quy trình thu thập dữ liệu (experience collection)
* Cấu hình replay buffer và sampling strategy
* Thiết lập parallel environments để tăng tốc huấn luyện
* Implement curriculum learning nếu cần thiết (từ dễ đến khó)

Monitoring và logging:

* Theo dõi các metrics: success rate, average reward, episode length
* Visualize learning curves và performance metrics
* Lưu trữ checkpoints và best models

Thiết lập benchmark:

* Xây dựng test scenarios chuẩn với các loại vật thể khác nhau
* Định nghĩa metrics đánh giá: success rate, completion time, stability score

So sánh thuật toán:

* Đánh giá hiệu quả của các thuật toán DRL khác nhau
* So sánh sample efficiency (số lượng mẫu cần để đạt hiệu suất tốt)
* Phân tích computational cost và inference time

Phân tích kết quả:

* Đánh giá khả năng generalization trên vật thể chưa thấy trong quá trình huấn luyện
* Phân tích failure cases và nguyên nhân
* Đề xuất hướng cải tiến

Nghiên cứu mô hình 6D pose:

* Tìm hiểu các phương pháp ước lượng tư thế 6D từ dữ liệu camera (RGB-D)
* Lựa chọn model phù hợp (ví dụ: DenseFusion, PVN3D, GDR-Net)

Tích hợp vào pipeline:

* Kết hợp thông tin 6D pose vào observation space
* Hỗ trợ cho việc định vị và lập kế hoạch grasping
  1. Đối tượng nghiên cứu
     1. Hệ thống robot

Nghiên cứu này tập trung vào bài toán điều khiển cánh tay robot kết hợp từ hai thành phần chính, tạo nên một hệ thống có bậc tự do (degrees of freedom) cao, rất phức tạp để điều khiển:

* **Cánh tay robot (Arm)**: Franka Emika FR3 - cánh tay robot 7 bậc tự do, chịu trách nhiệm di chuyển bàn tay đến vị trí của vật thể.
* **Bàn tay/Đầu cuối (End-effector)**: Shadow Hand - bàn tay robot hình người phức tạp, có nhiều ngón và khớp, cho phép thực hiện các thao tác cầm nắm tinh vi, mô phỏng lại sự khéo léo của bàn tay con người.
  + 1. Thuật toán điều khiển

Các thuật toán học tăng cường sâu (Deep Reinforcement Learning) được áp dụng để giải quyết bài toán điều khiển robot thực hiện tác vụ cầm nắm và nhấc vật thể.

* + 1. Môi trường mô phỏng

Không gian làm việc 3D được mô phỏng trong NVIDIA Isaac Lab, nơi robot và vật thể tương tác theo các định luật vật lý thực tế.

Các vật thể đa dạng về hình học được sinh ra ngẫu nhiên để thử thách khả năng tổng quát hóa của agent.

* 1. Phạm vi nghiên cứu

Nghiên cứu được triển khai hoàn toàn trong môi trường mô phỏng (simulation), không triển khai trên robot vật lý thật. Nền tảng mô phỏng chính được sử dụng là NVIDIA Isaac Lab.

Việc hiện thực hóa thuật toán và môi trường sử dụng ngôn ngữ lập trình Python và các thư viện liên quan (ví dụ: PyTorch).

Phạm vi bài toán:

* Tập trung vào tác vụ cầm, nắm và nhấc (grasping and lifting).
* Đầu vào cho hệ thống điều khiển giả định là đã có thông tin về trạng thái của robot và vị trí, tư thế (6D Pose) của vật thể. Nghiên cứu có thể tích hợp một mô hình ước lượng tư thế 6D nhưng không tập trung vào việc phát triển mô hình đó.

Phạm vi về yêu cầu hệ thống:

* Ưu tiên các thuật toán có khả năng huấn luyện và thực thi hiệu quả, hướng đến khả năng đáp ứng thời gian thực (real-time).
* Tập trung vào việc thiết kế hàm phần thưởng đơn giản, có khả năng khái quát hóa, hạn chế sự phụ thuộc vào kiến thức chuyên gia (domain knowledge) cho từng vật thể.
  1. Phương pháp nghiên cứu

Đề tài áp dụng phương pháp nghiên cứu thực nghiệm định lượng trong môi trường mô phỏng, kết hợp với phân tích so sánh để đánh giá hiệu quả của các thuật toán. Toàn bộ quy trình nghiên cứu được thực hiện theo các bước sau:

* + 1. Phương pháp Nghiên cứu Lý thuyết

Tổng quan và phân tích các công trình nghiên cứu liên quan đến việc ứng dụng học tăng cường sâu (Deep Reinforcement Learning - DRL) cho bài toán điều khiển và thao tác robot, đặc biệt là các tác vụ cầm nắm phức tạp (dexterous manipulation).

Nghiên cứu sâu về cơ chế hoạt động, ưu và nhược điểm của các thuật toán actor-critic tiêu biểu như SAC (off-policy, stochastic), DDPG (off-policy, deterministic) và các thuật toán on-policy như PPO.

Tìm hiểu các kỹ thuật cải thiện hiệu quả học mẫu (sample efficiency) như Hindsight Experience Replay (HER) và biến thể Relay Hindsight Experience Replay (RHER) trong điều khiển cánh tay robot.

* + 1. Phương pháp Mô phỏng (Simulation-based Approach)

Toàn bộ quá trình huấn luyện và kiểm thử được thực hiện trong môi trường mô phỏng vật lý NVIDIA Isaac Lab. Phương pháp này cho phép:

* Học song song và tăng tốc: Tận dụng sức mạnh của GPU để chạy nhiều môi trường song song, đẩy nhanh quá trình thu thập kinh nghiệm.
* An toàn và linh hoạt: Tránh được rủi ro hư hỏng phần cứng trên robot thật và dễ dàng thiết lập lại các kịch bản, vật thể.
* Cung cấp nền tảng cho Sim-to-Real: Các chính sách được huấn luyện trong môi trường mô phỏng có độ trung thực cao này là tiền đề để chuyển giao sang robot vật lý.
  + 1. Phương pháp Thực nghiệm So sánh (Comparative Experimental Method)

Đây là phương pháp cốt lõi của đề tài. Một loạt các thí nghiệm được thiết kế để so sánh hiệu suất giữa các phương pháp tiếp cận khác nhau trên cùng một bài toán và điều kiện môi trường. Cụ thể:

So sánh kiến trúc Actor-Critic: Đối chiếu giữa chính sách ngẫu nhiên (Stochastic - SAC) và chính sách tất định (Deterministic - DDPG).

So sánh chiến lược tối ưu hóa: Đối chiếu giữa các thuật toán Off-policy (SAC, DDPG) và thuật toán On-policy (PPO).

So sánh cơ chế học lại kinh nghiệm: Đối chiếu giữa Hindsight Experience Replay tiêu chuẩn (HER) và biến thể nâng cao (RHER) trên cùng một thuật toán nền (SAC và DDPG).

* + 1. Phương pháp Thiết kế Hàm thưởng (Reward Shaping)

Xây dựng một hàm thưởng có cấu trúc (structured reward function) để dẫn dắt tác nhân (agent) học hiệu quả hơn. Hàm thưởng được thiết kế dựa trên các thành phần được tính toán ở mỗi bước thời gian, bao gồm:

* Phần thưởng cho việc giảm khoảng cách từ các ngón tay đến vật thể.
* Phạt khi có va chạm giữa các ngón tay (self-collision).
* Phần thưởng lớn khi cầm nắm thành công và nhấc vật thể lên đạt một độ cao mục tiêu.
  + 1. Phương pháp Đánh giá Định lượng

Hiệu suất của mỗi thuật toán được đo lường và đánh giá dựa trên các số liệu định lượng cụ thể:

* Tỷ lệ thành công (Success Rate): Phần trăm số lần thực hiện tác vụ thành công trong một tập các thử nghiệm.
* Tốc độ hội tụ (Convergence Speed): Số lượng bước tương tác hoặc thời gian cần thiết để thuật toán đạt được một ngưỡng hiệu suất nhất định.
* Phần thưởng trung bình (Average Return): Phần thưởng tích lũy trung bình qua các episode, phản ánh chất lượng tổng thể của chính sách.
  1. Ý nghĩa khoa học và thực tiễn

**Về mặt khoa học**

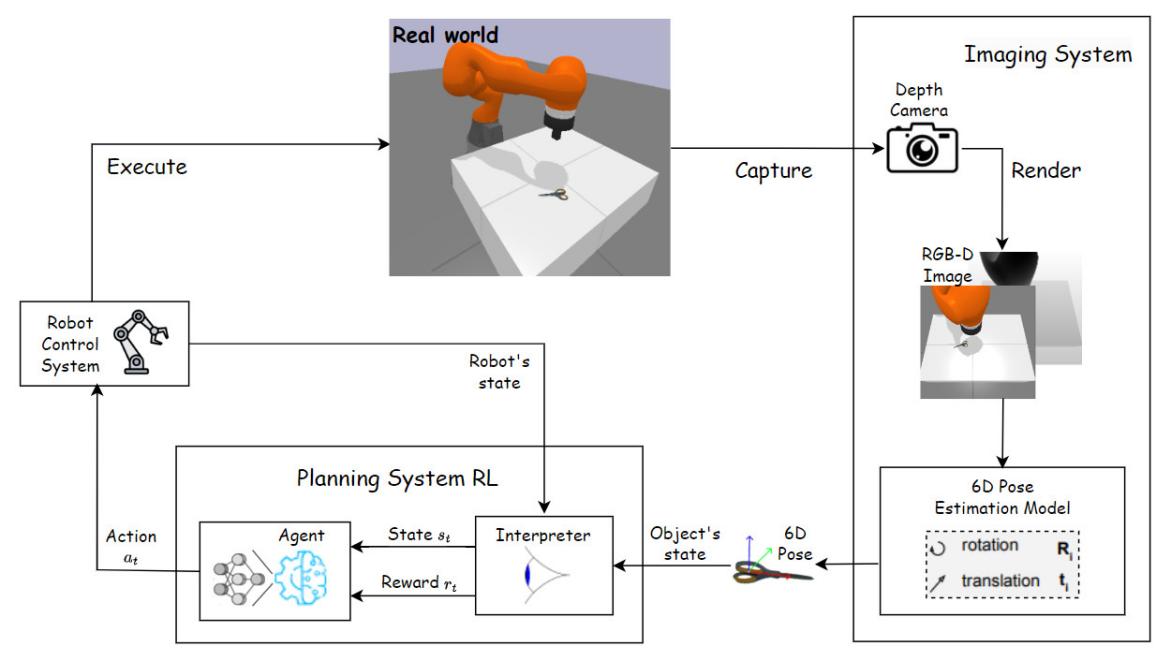
Đề tài góp phần vào việc **phát triển và cải tiến các thuật toán Deep Reinforcement Learning (Deep RL)** cho bài toán điều khiển robot trong môi trường liên tục. Việc nghiên cứu và ứng dụng các kỹ thuật tiên tiến không chỉ nâng cao hiệu quả học tập của robot mà còn **mở rộng phạm vi ứng dụng của Deep RL trong lĩnh vực robot,** cung cấp thêm bằng chứng thực nghiệm về tính hiệu quả của các kỹ thuật này, đóng góp vào sự phát triển chung của lĩnh vực trí tuệ nhân tạo và điều khiển robot.

**Về mặt thực tiễn:**

Đề tài này có tiềm năng ứng dụng cao trong thực tế, đặc biệt là trong lĩnh vực **tự động hóa**. Việc phát triển một hệ thống điều khiển robot thông minh, có khả năng thực hiện các thao tác phức tạp một cách tự động, sẽ mang lại nhiều lợi ích thiết thực như: n**âng cao năng suất lao động, cải thiện chất lượng sản phẩm, tiết kiệm chi phí, v.v**

1. CƠ SỞ LÝ THUYẾT
   1. Tổng quan
      1. Hệ thống điều khiển cánh tay robot truyền thống

Hệ thống điều khiển cánh tay robot truyền thống bao gồm ba thành phần chính: cánh tay robot (robotic arm), bộ gắp cuối (end-effector) và bộ điều khiển (controller) phối hợp chuyển động của cả hai [1]. Quy trình hoạt động điển hình của hệ thống được minh họa trong Hình 2.1.



Hình 2.1: Sơ đồ quy trình điều khiển cánh tay Robot

Các mô-đun chính tham gia vào hoạt động của cánh tay robot bao gồm:

**Mô-đun thị giác máy tính (Computer Vision Module**): Sử dụng camera chiều sâu (depth camera) hoặc camera RGB-D để thu thập thông tin không gian 3D của môi trường làm việc. Mô-đun này chịu trách nhiệm nhận dạng vật thể (object detection), phân đoạn (segmentation) và ước tính vị trí, tư thế 6D (6D pose estimation) của các vật thể trong không gian làm việc.

**Mô-đun học tập và lập kế hoạch (Learning and Planning Module):** Dựa trên thông tin thu được từ mô-đun thị giác, mô-đun này thực hiện việc lập kế hoạch chuyển động (motion planning), xác định chính sách điều khiển phù hợp (policy selection), và lựa chọn chuỗi hành động (action sequence) để đạt được mục tiêu mong muốn [2, 3].

**Mô-đun điều khiển (Control Module):** Chuyển đổi kế hoạch cấp cao thành các lệnh điều khiển cấp thấp (low-level commands) cho các khớp (joints) của robot, đảm bảo robot thực hiện chính xác quỹ đạo đã được lập kế hoạch.

Mặc dù hệ thống truyền thống đã chứng minh được hiệu quả trong các môi trường có cấu trúc và nhiệm vụ đơn giản, chúng vẫn tồn tại những hạn chế đáng kể:

* Thiếu khả năng thích ứng: Các hệ thống được lập trình cứng (hard-coded) gặp khó khăn khi đối mặt với các tình huống mới hoặc vật thể chưa được huấn luyện trước.
* Phụ thuộc vào kiến thức chuyên gia: Việc thiết kế bộ điều khiển và hàm phần thưởng đòi hỏi kiến thức chuyên môn sâu về động học (kinematics) và động lực học (dynamics) của robot.
* Khó mở rộng: Khi số lượng vật thể và độ phức tạp của nhiệm vụ tăng lên, việc duy trì và mở rộng hệ thống trở nên cực kỳ tốn kém.
  + 1. Bối cảnh và động lực nghiên cứu

Trong bối cảnh công nghệ phát triển vượt bậc, nhu cầu về các hệ thống robot thông minh, có khả năng tự học và thích ứng với môi trường động ngày càng trở nên cấp thiết. Đặc biệt trong các ứng dụng thực tế như:

* **Tự động hóa công nghiệp:** Robot cần xử lý nhiều loại sản phẩm khác nhau trên cùng một dây chuyền sản xuất.
* **Logistics và kho bãi:** Robot phải thao tác với các vật thể có hình dạng, kích thước và trọng lượng đa dạng.
* **Dịch vụ y tế:** Robot hỗ trợ phẫu thuật hoặc chăm sóc người cao tuổi cần độ chính xác và khả năng thích ứng cao.

Tuy nhiên, việc xây dựng một hệ thống robot "phổ quát" (general-purpose) gặp phải nhiều thách thức then chốt:

**Vấn đề phần thưởng thưa thớt (Sparse Reward Problem):** Trong nhiều tác vụ robot thực tế, tín hiệu phần thưởng chỉ xuất hiện khi hoàn thành nhiệm vụ (ví dụ: chỉ nhận phần thưởng khi vật thể được đặt đúng vị trí). Điều này khiến agent khó học được chiến lược hiệu quả từ các thử nghiệm ban đầu.

**Hiện tượng quên thảm họa (Catastrophic Forgetting):** Khi mô hình học một nhiệm vụ mới, nó có xu hướng quên đi những kiến thức đã học từ các nhiệm vụ trước đó, đặc biệt trong bối cảnh học đa nhiệm vụ (Multi-task Learning - MTL).

**Độ phức tạp của không gian trạng thái-hành động:** Với robot có nhiều bậc tự do (degrees of freedom - DoF) như hệ thống Franka FR3 (7-DoF) kết hợp Shadow Hand (24-DoF), không gian trạng thái và hành động trở nên cực kỳ lớn, tạo ra thách thức cho các thuật toán học truyền thống.

**Yêu cầu về sample efficiency:** Việc thu thập dữ liệu trên robot thật rất tốn kém về thời gian và chi phí, do đó thuật toán cần có khả năng học hiệu quả từ số lượng mẫu hạn chế.

* + 1. Phương pháp tiếp cận của nghiên cứu

Để giải quyết các thách thức nêu trên, nghiên cứu này đề xuất một phương pháp tiếp cận tích hợp, kết hợp ba lĩnh vực chính:

**Học tăng cường (Reinforcement Learning - RL):** Cung cấp framework cho phép agent học cách hành động tối ưu thông qua tương tác với môi trường, không cần lập trình trước các hành vi cụ thể.

**Học sâu (Deep Learning):** Sử dụng mạng nơ-ron sâu (Deep Neural Networks) để xấp xỉ các hàm phức tạp (chính sách, hàm giá trị) trong không gian trạng thái-hành động liên tục và chiều cao.

**Robot học (Robotics):** Cung cấp kiến thức về động học, động lực học và lý thuyết cầm nắm để thiết kế môi trường mô phỏng chân thực và hàm phần thưởng hiệu quả.

Sự giao thoa giữa ba lĩnh vực này tạo nên phương pháp Học tăng cường sâu (Deep Reinforcement Learning - DRL), là nền tảng cốt lõi của nghiên cứu.

A close up of a black and white background

AI-generated content may be incorrect.

Hình 2.2: Sự giao thoa giữa ba lĩnh vực tạo nên DRL

Cụ thể, nghiên cứu tập trung vào:

1. **Goal-Conditioned Reinforcement Learning (GCRL):** Mở rộng RL truyền thống bằng cách điều kiện hóa chính sách trên mục tiêu cụ thể, cho phép agent học cách đạt được nhiều mục tiêu khác nhau từ cùng một tập kinh nghiệm.
2. **Soft Actor-Critic (SAC):** Thuật toán off-policy hiện đại với khả năng sample efficiency cao, đặc biệt phù hợp với không gian hành động liên tục của robot.
3. **Relay Hindsight Experience Replay (RHER):** Kỹ thuật tiên tiến giúp giải quyết vấn đề sparse reward trong các nhiệm vụ tuần tự (sequential tasks) bằng cách relabel các mục tiêu đã đạt được.
4. **Multi-stage Task Decomposition:** Phân tách nhiệm vụ phức tạp thành các giai đoạn con (sub-tasks) với độ khó tăng dần, áp dụng curriculum learning để tăng tốc quá trình học.

Kiến trúc tổng thể của hệ thống được minh họa trong Hình 2.3:

A screenshot of a computer

AI-generated content may be incorrect.

Hình 2.3: Sự giao thoa giữa ba lĩnh vực tạo nên DRL

* 1. Cơ sở Robot học (Robotics)
     1. Động học robot (Kinematics)

Động học robot nghiên cứu mối quan hệ giữa vị trí các khớp (joint positions) và vị trí, tư thế của đầu cuối (end-effector pose) mà không xem xét lực và mô-men gây ra chuyển động. Đây là nền tảng quan trọng để điều khiển robot thực hiện các tác vụ thao tác.

* + - 1. Động học thuận (Forward Kinematics)

Động học thuận giải quyết bài toán: Cho biết góc của tất cả các khớp, hãy xác định vị trí và hướng của end-effector.

Đối với một cánh tay robot có n khớp, vị trí và hướng của end-effector được biểu diễn bằng ma trận biến đổi thuần nhất (homogeneous transformation matrix):

Trong đó:

* là ma trận biến đổi từ khung tọa độ thứ i sang khung tọa độ thứ i+1

* (Special Euclidean group) biểu diễn cả vị trí và hướng

Ma trận biến đổi có dạng:

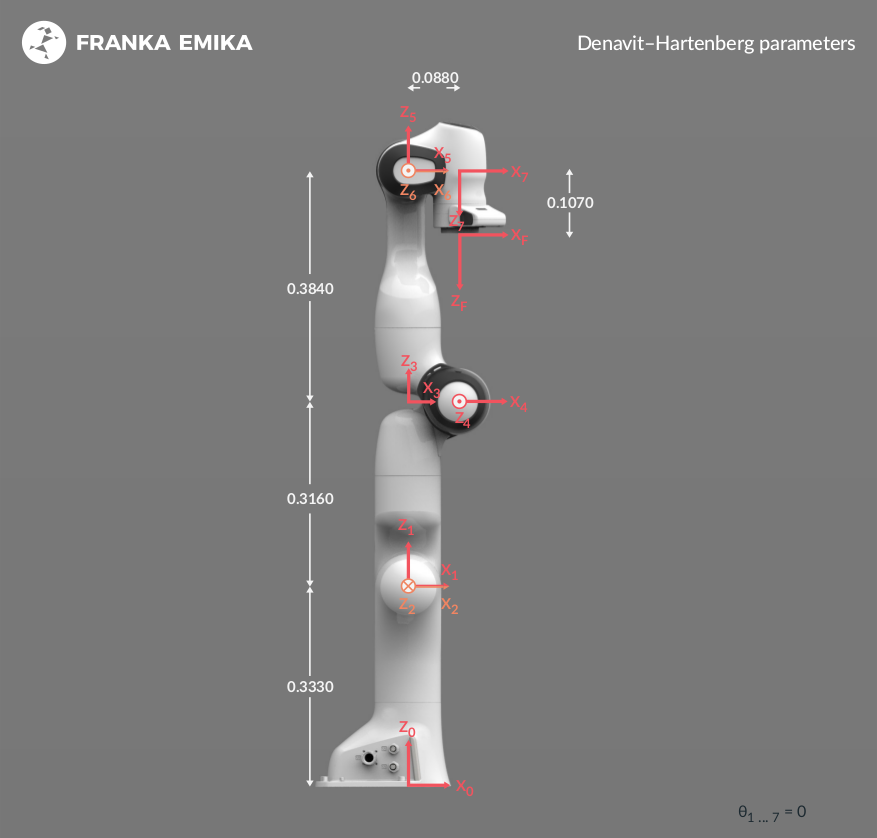
Trong đó:

* : Ma trận xoay 3×3 (rotation matrix)
* : Vector vị trí (position vector)

Phương pháp Denavit-Hartenberg (DH) thường được sử dụng để mô tả hệ tọa độ của các khớp, với bốn tham số cơ bản: .

Ví dụ với Franka Emika FR3:

Franka FR3 là cánh tay robot 7-DoF (7 bậc tự do), có cấu hình khớp như sau:



Hình 2.4: Cấu hình khớp của cánh tay Franka Emika FR3

Vị trí của end-effector được tính:

Trong đó  là vector góc khớp.

* + - 1. Động học ngược (Inverse Kinematics - IK)

Động học ngược giải quyết bài toán ngược lại: Cho biết vị trí và hướng mong muốn của end-effector, hãy tìm góc các khớp tương ứng.

Bài toán IK được phát biểu như sau:

Đối với robot 7-DoF như Franka FR3, bài toán IK có vô số nghiệm (redundant manipulator) do số bậc tự do lớn hơn số chiều của không gian tác vụ (task space) thông thường là 6 (3 vị trí + 3 hướng).

Phương pháp giải IK:

a. Phương pháp giải tích (Analytical methods):

* Phù hợp với robot có cấu trúc đặc biệt
* Cho nghiệm chính xác nhưng phức tạp

b. Phương pháp số (Numerical methods):

* Sử dụng Jacobian matrix:

Trong đó  là Jacobian matrix.

* Thuật toán lặp (iterative):

Trong đó  là pseudo-inverse của Jacobian.

c. Phương pháp học sâu:

* Sử dụng neural networks để học ánh xạ từ end-effector pose → joint angles
* Ưu điểm: nhanh, có thể xử lý ràng buộc phức tạp

Trong nghiên cứu này, việc sử dụng Deep RL cho phép agent tự học cách điều khiển mà không cần giải IK tường minh, giảm thiểu phụ thuộc vào mô hình toán học chính xác.

* + 1. Động lực học robot (Robot Dynamics)

Động lực học robot nghiên cứu mối quan hệ giữa lực/mô-men (forces/torques) tác động lên robot và chuyển động (gia tốc) của nó. Điều này rất quan trọng trong mô phỏng vật lý chân thực.

Phương trình động lực học của robot được mô tả bởi phương trình Euler-Lagrange:

Trong đó:

* : Ma trận quán tính (inertia matrix)
* : Ma trận Coriolis và lực ly tâm
* : Vector mô-men trọng lực (gravity torques)
* : Vector mô-men điều khiển (control torques)
* : Vị trí, vận tốc, gia tốc góc của các khớp

**Forward Dynamics:** Cho biết , tính : **Inverse Dynamics:** Cho biết , tính :

Trong môi trường mô phỏng NVIDIA Isaac Lab, engine vật lý tích hợp sẵn tính toán chính xác phương trình động lực học, bao gồm:

* Va chạm (collision detection and response)
* Ma sát (friction models)
* Contact dynamics (tương tác tiếp xúc)
  + 1. Lý thuyết cầm nắm (Grasping Theory)

Lý thuyết cầm nắm nghiên cứu về điều kiện để tạo ra một thế nắm ổn định (stable grasp) trên vật thể. Đây là nền tảng để thiết kế hàm phần thưởng và đánh giá thành công của tác vụ grasping.

* + - 1. Form Closure

Một thế nắm đạt form closure nếu vật thể không thể di chuyển dù chỉ một khoảng cực nhỏ mà không vi phạm ràng buộc tiếp xúc với bàn tay. Điều kiện form closure đảm bảo vật thể bị khóa hoàn toàn bởi các ngón tay.

Điều kiện toán học:

* Cần ít nhất  điểm tiếp xúc trong không gian n chiều
* Ví dụ: Trong không gian 3D, cần ít nhất 4 điểm tiếp xúc không đồng phẳng

* + - 1. Force Closure

Một thế nắm đạt force closure nếu tồn tại một tập hợp lực tiếp xúc có thể cân bằng bất kỳ lực và mô-men nào tác động lên vật thể.

Điều kiện toán học (dựa trên Grasp Matrix):   
Trong đó:

* : Grasp matrix (m là số điểm tiếp xúc)
* : Vector lực tại các điểm tiếp xúc

* : Wrench (lực + mô-men) bên ngoài

Force closure đạt được khi  có rank đầy đủ và tồn tại nghiệm  với tất cả các thành phần dương (lực đẩy).

* + - 1. Chất lượng của grasp (Grasp Quality Metrics)

Để đánh giá mức độ "tốt" của một thế nắm, nhiều chỉ số được đề xuất:

a. Ferrari-Canny Metric:

Trong đó  là force closure cone.

b. Volume of Grasp Wrench Space (GWS): Càng lớn càng tốt, thể hiện khả năng chống lại nhiễu loạn.

c. Minimum Singular Value của Grasp Matrix:

Trong nghiên cứu này, thay vì tính toán tường minh các chỉ số trên, chúng tôi thiết kế hàm phần thưởng khuyến khích agent tự học cách tạo ra thế nắm ổn định thông qua:

* Phần thưởng dương khi vật thể được nhấc lên thành công
* Phần thưởng âm khi vật thể bị rơi
* Phần thưởng trung gian dựa trên khoảng cách ngón tay - vật thể
  + 1. Mô hình hóa robot trong mô phỏng
       1. Khái niệm link và joint trong robot

Trong robot học, cấu trúc của robot được mô tả dưới dạng chuỗi động học (kinematic chain), bao gồm hai thành phần chính:

**Link:** các bộ phận dạng “thân” hoặc “đoạn nối” của robot, mang hình dạng vật lý(mesh), có khối lượng, quán tính và chiều dài xác định. Link không tự chuyển động mà được kết nối với nhau thông qua các khớp.

**Joint:** các khớp nối giữa các link, cho phép link chuyển động tương đối với nhau. Mỗi joint xác định một bậc tự do (Degree of Freedom – DOF), chẳng hạn:

* Revolute joint: khớp quay
* Prismatic joint: khớp trượt
* Fixed joint: khớp cố định

Số lượng joint quyết định độ linh hoạt của robot và được sử dụng trực tiếp trong điều khiển và mô phỏng động học.

* + - 1. Mô hình hóa robot trong mô phỏng Isaacsim

Isaac Sim sử dụng định dạng USD (Universal Scene Description) để mô tả hình học, động học và tính chất vật lý của robot.Cấu hình này bao gồm:

a. Mô hình hình học (Mesh)

* Được lấy từ file USD: Franka\_robot.usd
* Chứa tất cả link của cánh tay Franka và các ngón của Shadow Hand.
* Mỗi link bao gồm: mesh, mass, inertial tensor, collision shape.

b. Tham số động lực học

* Mass, inertia, damping, stiffness được định nghĩa cho từng nhóm khớp.
* Ví dụ:

panda\_arm = ImplicitActuatorCfg(

effort\_limit=87, velocity\_limit=1.7, stiffness=300, damping=80

)

c. Actuator và giới hạn khớp

Mỗi joint có:

* Giới hạn góc (joint limit)
* Giới hạn vận tốc (velocity\_limit)
* Giới hạn mô-men/ lực (effort\_limit)

Điều này đảm bảo mô phỏng bám sát robot thật.

* + - 1. Quy trình đưa robot từ đời thực vào mô phỏng

Quy trình chuẩn để đưa một robot thật (ví dụ Franka + Shadow Hand) vào mô phỏng Isaac Sim thường gồm các bước:

Bước 1: Chuẩn bị mô hình hình học

* Lấy mesh từ CAD hoặc URDF của robot thật.
* Chuyển đổi sang định dạng USD hoặc mở trực tiếp bằng Isaac Sim.

Bước 2: Xác định cấu trúc động học

* Liệt kê tất cả link và joint theo cây động học.
* Xác định loại joint (quay/trượt/cố định).
* Gán bậc tự do (DOF) cho từng khớp.

Bước 3: Thiết lập thuộc tính vật lý

* Gán mass, inertia cho từng link.
* Xác định collision shape phù hợp.
* Thiết lập ma sát, damping, stiffness.

Bước 4: Tạo file robot USD

* Gom toàn bộ mesh + link + joint thành 1 file USD duy nhất.
* Gắn các metadata cần thiết (joints, limits, physic material).

Bước 5: Định nghĩa ArticulationCfg trong Isaac Lab

* Chỉ rõ file USD nguồn.
* Khai báo joint\_pos, actuator, giới hạn lực, tốc độ.
* Tổ chức các nhóm khớp (arm, wrist, từng ngón tay).

Bước 6: Kiểm tra mô phỏng

* Spawn robot trong một scene rỗng.
* Kiểm thử:
  + joint di chuyển đúng không
  + link không bị xuyên
  + collision chính xác
  + actuator hoạt động ổn định

Bước 7: Kết hợp với môi trường RL

* Tạo observation (pose, joint state).
* Tạo action mapping theo số DOF.
* Tích hợp vào Isaac Lab để huấn luyện multi-env.
  + 1. Cấu tạo hệ thống robot sử dụng

Nghiên cứu này sử dụng một hệ thống robot phức hợp có tổng cộng 31 bậc tự do (DoF), bao gồm:

**Franka Emika FR3 (7-DoF Arm)**

Thông số kỹ thuật:

* Số khớp: 7 khớp xoay (revolute joints)
* Payload: 3 kg
* Reach: 855 mm
* Repeatability: ±0.1 mm
* Joint limits: Được thiết kế để tránh singularities

**Cấu trúc khớp:**

Bảng 2.1: Đặc tính các khớp của robot Franka Emika FR3

| **Joint** | **Loại khớp** | **Giới hạn** | **Vận tốc tối đa** |
| --- | --- | --- | --- |
| 1 | Revolute | ±166° | 150°/s |
| 2 | Revolute | ±101° | 150°/s |
| 3 | Revolute | ±166° | 150°/s |
| 4 | Revolute | ±176° | 150°/s |
| 5 | Revolute | ±166° | 180°/s |
| 6 | Revolute | ±1° | 180°/s |
| 7 | Revolute | ±166° | 180°/s |

**Ưu điểm:**

* Dư thừa động học (kinematic redundancy) với 7-DoF cho phép tránh chướng ngại vật
* Độ chính xác cao
* Tích hợp sẵn cảm biến mô-men tại từng khớp

**Shadow Dexterous Hand (24-DoF)**

Shadow Hand là bàn tay robot mô phỏng bàn tay người, có khả năng thực hiện các thao tác tinh vi.

Bảng 2.2: Đặc tính các khớp của bàn tay Shadow Hand

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Bộ phận** | **Số joint** | **Tên khớp** | **Loại khớp** | **Giới hạn** | **Vận tốc tối đa** |
| Cổ tay - Wrist joint | 2 | WRJ0 | Revolute | ±30° | 180°/s |
| WRJ1 | Revolute | ±45° | 180°/s |
| Ngón cái - Thumb | 5 | THJ0 | Revolute | ±20° | 200°/s |
| THJ1 | Revolute | ±20° | 200°/s |
| THJ2 | Revolute | ±15° | 200°/s |
| THJ3 | Revolute | ±15° | 200°/s |
| THJ4 | Revolute | ±12° | 200°/s |
| Ngón trỏ - First finger | 4 | FFJ0 | Revolute | ±15° | 200°/s |
| FFJ1 | Revolute | ±25° | 200°/s |
| FFJ2 | Revolute | ±25° | 200°/s |
| FFJ3 | Revolute | ±25° | 200°/s |
| **Bộ phận** | **Số joint** | **Tên khớp** | **Loại khớp** | **Giới hạn** | **Vận tốc tối đa** |
| Ngón giữa – Middle finger | 4 | MFJ0 | Revolute | ±15° | 200°/s |
| MFJ1 | Revolute | ±25° | 200°/s |
| MFJ2 | Revolute | ±25° | 200°/s |
| MFJ3 | Revolute | ±25° | 200°/s |
| Ngón áp út – Ring finger | 4 | RFJ0 | Revolute | ±15° | 200°/s |
| RFJ1 | Revolute | ±25° | 200°/s |
| RFJ2 | Revolute | ±25° | 200°/s |
| RFJ3 | Revolute | ±25° | 200°/s |
| Ngón út – Little finger | 5 | LFJ0 | Revolute | ±15° | 200°/s |
| LFJ1 | Revolute | ±25° | 200°/s |
| LFJ2 | Revolute | ±25° | 200°/s |
| LFJ3 | Revolute | ±25° | 200°/s |
| LFJ4 | Revolute | ±12° | 200°/s |

Các khớp của mỗi ngón (trừ thumb):

1. MCP (Metacarpophalangeal**):** 2 DoF (flexion/extension, abduction/adduction)
2. PIP (Proximal Interphalangeal): 1 DoF (flexion/extension)
3. DIP (Distal Interphalangeal): 1 DoF (flexion/extension, coupled với PIP)

Không gian hành động của toàn bộ hệ thống là:

Trong đó mỗi chiều tương ứng với target position hoặc target torque cho một khớp.

Thách thức điều khiển:

* High-dimensional action space: 31 chiều làm tăng độ phức tạp của bài toán tối ưu
* Coordination**:** Cần điều phối đồng bộ giữa arm (di chuyển) và hand (grasping)
* Contact-rich manipulation: Nhiều điểm tiếp xúc đồng thời giữa ngón tay và vật thể
* Underactuation: Một số khớp của Shadow Hand được điều khiển gián tiếp qua dây cáp (tendon-driven)
  1. Học tăng cường (Reinforcement Learning)
     1. Giới thiệu và khái niệm cơ bản

Học tăng cường (Reinforcement Learning - RL) là một paradigm của học máy, trong đó một agent học cách hành xử tối ưu trong một môi trường (environment) thông qua tương tác thử-và-sai (trial-and-error). Khác với học có giám sát (supervised learning) - nơi mô hình học từ các cặp (input, label) được gán nhãn sẵn, RL không có "giáo viên" trực tiếp mà chỉ có tín hiệu phần thưởng (reward signal) cho biết hành động tốt hay xấu.

Đặc điểm chính của RL:

1. Sequential decision making: Agent đưa ra chuỗi quyết định liên tiếp, mỗi quyết định ảnh hưởng đến trạng thái tương lai
2. Delayed rewards: Hậu quả của một hành động có thể không rõ ràng ngay lập tức
3. Exploration vs. Exploitation tradeoff:

* Exploration: Thử các hành động mới để khám phá môi trường
* Exploitation: Sử dụng kiến thức đã có để tối đa hóa phần thưởng

1. Credit assignment problem**:** Xác định hành động nào trong quá khứ đóng góp vào thành công/thất bại hiện tại

**Ví dụ minh họa:** Dạy robot chơi cờ vua

* **State:** Bàn cờ hiện tại
* **Action:** Nước đi tiếp theo
* **Reward:** +1 nếu thắng, -1 nếu thua, 0 nếu hòa (sparse reward)
* Agent phải học chiến lược (policy) để tối đa hóa xác suất thắng
  + 1. Các thành phần cốt lõi

Một hệ thống RL bao gồm các thành phần sau:

Agent (Tác nhân)

Agent là thực thể ra quyết định, đại diện cho hệ thống điều khiển cần được huấn luyện. Trong bài toán của chúng ta:

* **Agent = Policy network** (mạng chính sách) điều khiển robot
* Nhận đầu vào: Trạng thái hiện tại  (vị trí khớp, vị trí vật thể, mục tiêu...)
* Đầu ra: Hành động  (target positions/torques cho các khớp)

Environment (Môi trường)

Environment là "thế giới" mà agent tương tác. Nó phản hồi lại hành động của agent bằng:

* Trạng thái mới
* Phần thưởng

Trong nghiên cứu này:

* Environment = NVIDIA Isaac Lab simulation
  + Mô phỏng vật lý: Franka FR3 + Shadow Hand + Objects + Table
  + Physics engine: Tính toán dynamics, collisions, contacts
  + Sensor simulation: Joint sensors, contact sensors, cameras (nếu có)

State (Trạng thái) –

State là biểu diễn đầy đủ về tình trạng hiện tại của môi trường tại thời điểm .

Trong bài toán robot manipulation, state bao gồm:

Trong đó:

* : Observation (quan sát)

* : Goal (mục tiêu)

Chi tiết observation :

Trong đó:

* : Góc khớp của Franka arm

* : Vận tốc góc của Franka arm

* : Góc khớp của Shadow Hand

* : Vận tốc góc của Shadow Hand

* : Vị trí 3D của vật thể

* (hoặc quaternion ): Hướng của vật thể

* : Vận tốc tịnh tiến của vật thể

* : Vận tốc góc của vật thể

Tổng số chiều của observation:  chiều

Tính chất Markov:

State phải thỏa mãn Markov Property:

Nghĩa là: "Tương lai chỉ phụ thuộc vào hiện tại, không phụ thuộc vào quá khứ"

Trong thực tế, đôi khi observation không đầy đủ (Partially Observable MDP - POMDP), khi đó cần sử dụng recurrent networks (LSTM, GRU) để duy trì memory về quá khứ. Nghiên cứu này sử dụng **SeqGRUNet** để xử lý sequence.

Action (Hành động) –

Action là lệnh điều khiển mà agent gửi đến robot.

Continuous action space:

Mỗi chiều tương ứng với:

* **Position control:** Target joint position
* **Torque control:** Target joint torque

Bounded action space:

Trong thực tế, actions bị giới hạn:

Ví dụ với position control:

Trong implementation, actions thường được chuẩn hóa về  và scale lại khi áp dụng:

Reward (Phần thưởng) –

Reward là tín hiệu số cho biết "độ tốt" của hành động  tại trạng thái .

Rewardshaping (thiết kế hàm phần thưởng) là nghệ thuật quan trọng trong RL.

Sparse vs. Dense Rewards:

**1. Sparse Reward:** (Được sử dụng trong nghiên cứu này)

Ví dụ cho grasp task:

Ưu điểm:

* Đơn giản, không cần kiến thức chuyên gia
* Tránh exploitation của các heuristics không mong muốn

Nhược điểm:

* Khó học do thiếu guidance
* Cần kỹ thuật đặc biệt như HER, RHER

2. Dense (Shaped) Reward:

Ví dụ:

Ưu điểm**:** Học nhanh hơn do có guidance liên tục

Nhược điểm:

* Cần expert knowledge
* Có thể dẫn đến unexpected behaviors (reward hacking)

Multi-stage Reward trong nghiên cứu này:

Để cân bằng giữa simplicity và learning efficiency, chúng tôi sử dụng multi-stage sparse rewards:

**Stage 0 (Reach):**

**Stage 1 (Lift):**

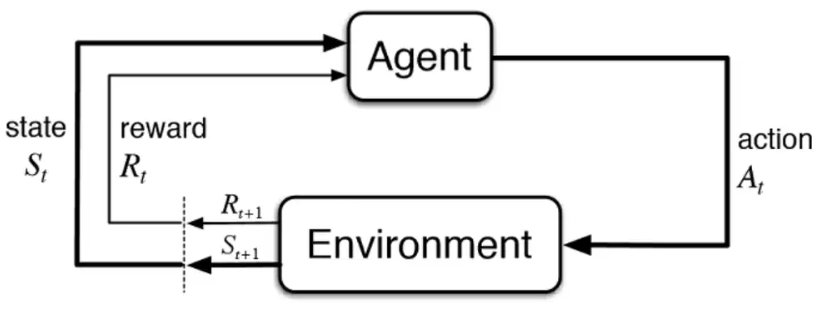
Điều này cho phép curriculum learning: học task đơn giản (reach) trước, sau đó chuyển sang task phức tạp (lift).

* + 1. Quy trình ra quyết định Markov (Markov Decision Process - MDP)

Các bài toán học tăng cường thường được mô tả dưới dạng Quy trình quyết định Markov (MDP), được xác định bởi sáu yếu tố , trong đó:

* là tập hợp các trạng thái
  + Có thể rời rạc (discrete):
  + Hoặc liên tục (continuous):  (như trong bài toán robot)
* : **Action space** - Tập hợp tất cả các hành động có thể
  + Discrete:  (ví dụ: lên, xuống, trái, phải)
  + Continuous:  (như trong điều khiển robot)
* *:*Phân phối trạng thái ban đầu
* là phân phối xác suất chuyển tiếp hay còn được gọi là dynamic model
* là hàm phần thưởng hoặc đơn giản hơn:
* là hệ số chiết khấu
  + : Chỉ quan tâm phần thưởng tức thời (myopic)
  + : Coi tất cả phần thưởng tương lai như nhau
  + Thường:

Vào đầu mỗi tập, một trạng thái ban đầu được lấy mẫu từ phân phối . Tại bước thời gian t, tác nhân ở trạng thái và quyết định thực hiện hành động theo chính sách , . Tại bước thời gian tiếp theo t+1, nó chuyển sang trạng thái được lấy mẫu từ phân phối và nhận được phần thưởng . Mục tiêu của tác nhân trong RL là tối đa hóa phần thưởng tích lũy ​ bằng cách khám phá các trạng thái khác nhau và học chính sách . Một cách trực quan, Hình 10 dưới đây sẽ minh họa rõ hơn về quy trình tương tác giữa tác tử và môi trường trong học tăng cường.



Hình 2.5: Sơ đồ về cách thức hoạt động của học tăng cường (RL)

**Trajectory (quỹ đạo):** Một chuỗi các trạng thái, hành động, phần thưởng:

**Policy (Chính sách):** Policy  là chiến lược hành động của agent, ánh xạ từ trạng thái sang hành động.

Stochastic policy:

Ví dụ:  (Gaussian policy)

Deterministic policy:

Ví dụ:

Trong DRL, policy thường được tham số hóa bởi neural network với tham số :

Return (Lợi nhuận tích lũy)

Return là tổng phần thưởng chiết khấu từ thời điểm :

Với episode hữu hạn (kết thúc tại bước ):

**Ý nghĩa của :**

* gần 0: Ưu tiên phần thưởng ngắn hạn
* gần 1: Coi trọng phần thưởng dài hạn
* đảm bảo  hội tụ và tránh infinite sums

Value Functions (Hàm giá trị)

Value functions ước tính "độ tốt" của trạng thái hoặc cặp trạng thái-hành động.

1. State-Value Function :

Giá trị kỳ vọng của return khi bắt đầu từ trạng thái  và theo chính sách :

Phương trình Bellman cho :

**2. Action-Value Function :**

Giá trị kỳ vọng khi thực hiện hành động  tại , sau đó theo chính sách :

**Phương trình Bellman cho :**

Mối quan hệ giữa  và :

**Optimal Policy và Optimal Value Functions**

**Mục tiêu của RL:** Tìm chính sách tối ưu  tối đa hóa expected return:

**Optimal state-value function:**

Optimal action-value function: Bellman Optimality Equation cho :

Bellman Optimality Equation cho :

Optimal policy từ :

Greedy policy với respect to :

**Temporal Difference (TD) Error:**

TD error đo lường sự khác biệt giữa estimate hiện tại và target mới.

**Advantage Function:**   
Advantage function cho biết hành động  tốt hơn (hoặc tệ hơn) bao nhiêu so với hành động trung bình tại trạng thái .

* : Hành động  tốt hơn average
* : Hành động  tệ hơn average

Advantage được sử dụng rộng rãi trong các thuật toán policy gradient hiện đại (PPO, A3C, SAC.. .).

* 1. Học Sâu (Deep Learning) và Học Tăng Cường Sâu (DRL
     1. Từ RL đến DRL: Vai trò của Deep Neural Networks

Các thuật toán RL cổ điển (như Q-learning, SARSA) sử dụng tabular methods - lưu trữ value function trong bảng tra cứu (lookup table):

| State | Action | Q-value |
| --- | --- | --- |
| s₁ | a₁ | 2.5 |
| s₁ | a₂ | 3.1 |
| s₂ | a₁ | 1.8 |
| ... | ... | ... |

Vấn đề:

1. Curse of dimensionality**:** Với không gian trạng thái liên tục hoặc chiều cao, số lượng states có thể là vô hạn hoặc cực kỳ lớn.

Ví dụ: Trong bài toán robot với observation , nếu rời rạc hóa mỗi chiều thành 10 giá trị →  states!

1. Không generalize**:** Tabular methods không thể khái quát hóa (generalize) từ states đã thấy sang states mới tương tự.

Giải pháp: Function Approximation

Thay vì lưu trữ giá trị cho từng state, ta sử dụng một **hàm tham số hóa** (parameterized function) để xấp xỉ:

Trong đó  là tham số của hàm xấp xỉ.

Deep Learning = Powerful Function Approximators

Deep Neural Networks (DNNs) là universal function approximators có khả năng:

* Học các biểu diễn phức tạp (complex representations)
* Generalize tốt từ dữ liệu huấn luyện
* Xử lý high-dimensional inputs (images, continuous states)

Deep Reinforcement Learning (DRL) = RL + Deep Learning

* State s → Neural Network → Action a (with params θ)
* State s → Neural Network → Value (with params φ)
  + 1. Mạng nơ-ron trong DRL
       1. Value Function Approximation

Sử dụng neural network để xấp xỉ value function:

1. Deep Q-Network (DQN):

Hoặc:

Trong đó output layer có  nodes, mỗi node tương ứng với .

Loss function (Mean Squared Bellman Error):

Trong đó:

* (TD target)

* : Target network parameters (frozen hoặc slowly updated)
* : Replay buffer

2. Value Network cho continuous state:

)

* + - 1. Policy Function Approximation

Sử dụng neural network để xấp xỉ policy:

**1. Discrete action space:** Output layer: logits

**2. Continuous action space:** (Sử dụng trong nghiên cứu này)

**Gaussian policy:** Trong đó:

* : Mean

* (hoặc diagonal covariance): Standard deviation

Architecture ví dụ:

|  |
| --- |
| Input: s ∈ ℝ⁷⁰  ↓  Shared layers:  Dense(256, SiLU)  Dense(256, SiLU)  ↓  ├──→ Dense(31) → μ(s)  │  └──→ Dense(31) → log σ(s) |

Sampling action:

Với Gaussian: , trong đó

**Reparameterization trick:**

Cho phép backpropagation qua stochastic sampling.

Squashing function (Tanh):

Để đảm bảo actions nằm trong bounds : Sau đó scale:

* + 1. Tổng quan các thuật toán DRL cho điều khiển robot liên tục

A diagram of a algorithm

AI-generated content may be incorrect.

Hình 2.6: Phân loại thuật toán DRL

Học tăng cường sâutrong các bài toán điều khiển liên tục cho robot là một phương pháp đầy hứa hẹn đã giải quyết thành công nhiều thách thức phức tạp và đạt được những cột mốc quan trọng.

* **Yang và cộng sự** [35] đã đề xuất phương pháp Học tăng cường sâu phân cấp, cho phép robot học cả kỹ năng cơ bản và kỹ năng phức hợp thông qua hệ thống phân cấp hai cấp.
* **Gao và cộng sự** [36] đã đề xuất bộ mã hóa dựa trên mạng LSTM để thu thập thông tin mới nhất về các chướng ngại vật, kết hợp với thuật toán deep deterministic policy gradient (DDPG) cải tiến để đạt được quỹ đạo không va chạm với các chướng ngại vật động.
* Trong môi trường thay đổi nhanh chóng, **Cheng và cộng sự** [37] đã đề xuất tăng cường chính sách RL với bộ điều khiển thích ứng L1 để cho phép ước lượng nhanh và bù trừ chủ động cho các biến thể động.
* **Shukla và cộng sự** [38] và **Pohlmeyer và cộng sự** [39] cũng đã đề xuất kiến trúc Actor-Critic để học các thay đổi môi trường mà không yêu cầu kiến thức trước về mô hình môi trường.

Mặc dù có nhiều tiến bộ, các phương pháp Deep RL hiện tại vẫn còn một số hạn chế:

* **Soft Actor-Critic** [16]: Chỉ hỗ trợ điều khiển robot trong điều kiện di chuyển an toàn, tập trung vào việc tránh va chạm.
* **TD3 kết hợp với HER** [40]: Tạo ra quỹ đạo mượt mà hơn, không va chạm, nhưng chưa giải quyết triệt để vấn đề hiệu quả mẫu.

Để giải quyết bài toán điều khiển cánh tay robot trong môi trường có phần thưởng thưa thớt (sparse rewards), nhiều thuật toán học tăng cường đã được nghiên cứu và áp dụng. Các thuật toán nền tảng như **Proximal Policy Optimization (PPO)**, **Deep Deterministic Policy Gradient (DDPG)**, và **Soft Actor-Critic (SAC)** đã chứng tỏ được hiệu quả nhất định. Đặc biệt, các phương pháp off-policy như DDPG và SAC cho thấy ưu thế về hiệu quả sử dụng mẫu (sample efficiency), một yếu tố then chốt trong các bài toán robot thực tế.

* + - 1. PPO (Proximal Policy Optimization)

PPO là thuật toán on-policy phổ biến nhất hiện nay.

Ưu điểm:

* Ổn định, dễ tune hyperparameters
* Hoạt động tốt trên nhiều tasks khác nhau
* Sample complexity tốt hơn TRPO

Nhược điểm:

* On-policy: Phải discard data sau mỗi lần update → kém sample-efficient
* Chậm hơn off-policy methods trong robot tasks

Objective function:

Trong đó:

* (probability ratio)

* : Estimated advantage
* : Clipping parameter
  + - 1. DDPG (Deep Deterministic Policy Gradient)

DDPG là thuật toán off-policy cho continuous action space.

Ưu điểm:

* Off-policy → có thể reuse data → sample-efficient
* Deterministic policy → dễ optimize

Nhược điểm:

* Kém ổn định, nhạy với hyperparameters
* Dễ bị overestimation bias trong Q-learning
* Exploration yếu (cần thêm noise)

Components:

1. Actor:  - deterministic policy
2. Critic:  - action-value function

Actor update (Policy Gradient):

Critic update (Bellman error minimization):

Trong đó

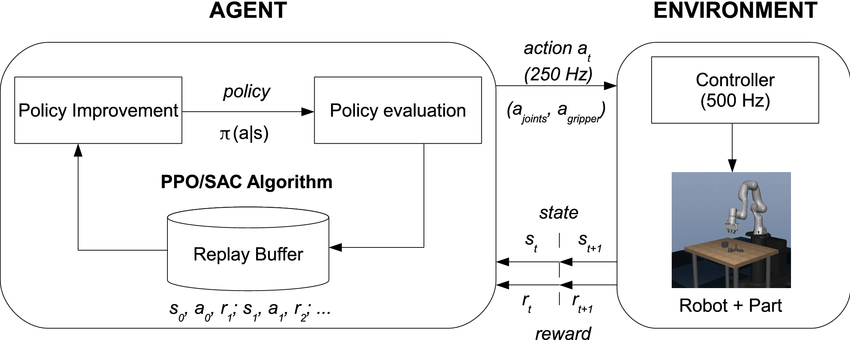
Để cải thiện hơn nữa khả năng học từ các tác vụ thất bại, một số kỹ thuật và thuật toán cải tiến đã ra đời:

**Hindsight Experience Replay (HER)**: Một kỹ thuật đột phá giúp tăng tốc độ hội tụ bằng cách "tưởng tượng" lại các mục tiêu đã đạt được từ những quỹ đạo (trajectories) thất bại. HER thường được kết hợp với các thuật toán off-policy, tạo ra các biến thể như DDPG+HER và SAC+HER.

**Relay Hindsight Experience Replay (RHER)**: Là một phiên bản mở rộng của HER, được thiết kế đặc biệt để giải quyết các hạn chế trong những nhiệm vụ có tính tuần tự (sequential tasks). RHER giúp cải thiện hiệu quả mẫu và tăng cường tính ổn định bằng cách tạo ra dữ liệu huấn luyện cho các bước trung gian.

Trong nghiên cứu này, chúng tôi tập trung vào việc tận dụng các phương pháp tiên tiến nhất để giải quyết bài toán điều khiển robot đa nhiệm phức tạp. Chúng tôi đề xuất sử dụng thuật toán SAC kết hợp với RHER (SAC+RHER) làm phương pháp chính. Sự kết hợp này được kỳ vọng sẽ tận dụng được khả năng khám phá mạnh mẽ và tính ổn định của SAC, đồng thời thừa hưởng hiệu quả vượt trội của RHER trong các nhiệm vụ tuần tự với phần thưởng thưa thớt.

Để chứng minh tính hiệu quả của phương pháp đề xuất, chúng tôi sẽ tiến hành so sánh hiệu suất của SAC+RHER với một loạt các thuật toán nền tảng và cải tiến khác, bao gồm PPO, DDPG, SAC, DDPG+HER, DDPG+RHER, và SAC+HER. Các kết quả thực nghiệm sẽ minh họa chi tiết ưu điểm của phương pháp đề xuất trong việc giải quyết bài toán điều khiển cánh tay robot.



Hình 2.7: Luồng xử lý PPO/SAC

* + 1. Các thuật toán Actor-Critic
       1. Kiến trúc Actor-Critic

Actor-Critic kết hợp ưu điểm của cả value-based và policy-based methods:

A diagram of a process

AI-generated content may be incorrect.

Hình 2.8: Actor Critic Process

Vai trò:

* Actor (Policy network): Chọn hành động
  + Input: State
  + Output: Action distribution  hoặc deterministic action
  + Learning signal: Được cung cấp bởi Critic
* Critic (Value network): Đánh giá hành động
  + Input: State  (và Action  nếu là Q-function)
  + Output: Value estimate  hoặc
  + Learning signal: TD error hoặc Bellman error

Luồng cập nhật:

1. Actor chọn action
2. Environment trả về
3. Critic tính TD error:
4. Update Critic: Giảm TD error
5. Update Actor: Tối đa hóa value estimate từ Critic
   * + 1. Actor (Policy Network)

Trong nghiên cứu này, chúng tôi sử dụng SeqGRUNet (sẽ chi tiết ở mục 2.10.1).

Policy Gradient update:

Trong đó  là advantage estimate từ Critic.

Với Gaussian policy:

* + - 1. Critic (Value Network)

Critic update (MSE loss):

Hoặc với Q-function:

* 1. Goal-Conditioned Reinforcement Learning (GCRL)
     1. Động lực và khái niệm

Trong RL truyền thống, agent học một chính sách duy nhất cho một nhiệm vụ cụ thể. Tuy nhiên, trong thực tế, robot thường phải thực hiện nhiều nhiệm vụ khác nhau hoặc đạt được nhiều mục tiêu khác nhau.

Ví dụ:

* Đặt vật thể A vào vị trí X
* Đặt vật thể B vào vị trí Y
* Đặt vật thể C vào vị trí Z

Thay vì huấn luyện 3 policies riêng biệt, **Goal-Conditioned RL (GCRL)** học một **universal policy** có thể đạt được nhiều goals khác nhau.

Ý tưởng cốt lõi:

Mở rộng state space để bao gồm cả **goal** (mục tiêu):

Trong đó:

* : Observation (quan sát về môi trường)
* : Goal (mô tả mục tiêu mong muốn)

Policy được điều kiện hóa trên cả observation và goal:

* + 1. Formulation của GCRL

Theo formulation của HER [48], GCRL MDP được mở rộng:

**Các thành phần mới:**

* : Goal space - Không gian các mục tiêu có thể

Ví dụ:  (vị trí 3D mục tiêu của vật thể)

* Augmented state:

Trong đó:

* : Observation

* : Desired goal (mục tiêu mong muốn)
* Achieved goal function:

Hàm  trích xuất achieved goal từ observation.

Ví dụ:

* chứa vị trí vật thể hiện tại:
* (chỉ lấy vị trí)

* : Mục tiêu "đã đạt được" tại state hiện tại

Goal-conditioned policy:

Goal-conditioned value functions:

Goal-conditioned reward function:

Thường được định nghĩa dựa trên khoảng cách giữa achieved goal và desired goal:

Hoặc:

* + 1. Ưu điểm trong bài toán đa nhiệm vụ và đa mục tiêu

1. Học một policy universal: Một policy duy nhất  có thể xử lý nhiều goals khác nhau → Multi-task learning

2. Sample efficiency:

Cùng một trajectory có thể được sử dụng để học cho nhiều goals khác nhau thông qua Hindsight Experience Replay (HER).

Ví dụ minh họa HER:

Giả sử robot muốn đặt vật thể vào vị trí  nhưng thất bại, chỉ đặt được vào .

Trong RL truyền thống: Episode này hoàn toàn thất bại, reward = 0 → không học được gì.

Trong GCRL + HER: Chúng ta "tưởng tượng lại" rằng mục tiêu ban đầu là  → Episode này thành công! → Học được cách đạt goal .

3. Generalization: Policy học được cách đạt các goals chưa thấy trong training (nội suy giữa các goals đã học).

4. Curriculum learning tự nhiên: Khi kết hợp với HER, agent tự động tạo curriculum từ dễ → khó bằng cách học các goals gần trước, xa sau.

* + 1. Áp dụng trong nghiên cứu này

Trong bài toán grasp and lift, chúng tôi định nghĩa:

**Observation :**

**Achieved goal :**  (Vị trí và hướng hiện tại của vật thể)

**Desired goal :**  (Vị trí và hướng mục tiêu của vật thể)

Multi-stage goals:

* **Stage 0 (Reach):**  (tiếp cận vật thể)
* **Stage 1 (Lift):**  (nhấc lên độ cao mong muốn)

Reward function:

* 1. Thuật toán Soft Actor-Critic (SAC)
     1. Maximum Entropy Reinforcement Learning

Soft Actor-Critic (SAC) [16] là một thuật toán off-policy actor-critic hiện đại, được xây dựng dựa trên framework Maximum Entropy Reinforcement Learning.

Ý tưởng cốt lõi:

Thay vì chỉ tối đa hóa expected return, SAC tối đa hóa entropy-augmented objective:

Trong đó:

* *:* Phần thưởng thông thường
* : Entropy của policy
* *:* Temperature parameter (hệ số entropy)

Entropy của phân phối:

Đối với Gaussian policy :

Entropy cao → Policy stochastic (ngẫu nhiên) → Khuyến khích **exploration**

Entropy thấp → Policy deterministic → Khuyến khích **exploitation**

Lợi ích của Maximum Entropy framework:

1. Automatic exploration: Policy tự động khám phá do cần duy trì entropy cao
2. Robustness: Policy stochastic ít bị overfitting vào một chiến lược cụ thể
3. Multi-modal solutions: Có thể tìm nhiều cách giải quyết task
4. Faster learning: Exploration tốt hơn → convergence nhanh hơn
   * 1. Entropy Regularization

Soft Q-function:

Định nghĩa lại Q-function với entropy term:

Trong đó soft value function:

Hoặc viết tường minh:

Soft Bellman Equation:

Optimal soft Q-function:

Trong đó:

Đây là **soft maximum** (log-sum-exp):

* + 1. Công thức toán học của SAC

SAC gồm ba components chính được cập nhật xen kẽ:

1. Critic Update (Twin Q-networks)

SAC sử dụng **hai Q-networks**  (clipped double Q-learning) để giảm overestimation bias.

**Loss function cho mỗi Q-network:**

Trong đó soft Bellman target:

Với:

* : Action được sample từ policy hiện tại
* : Terminal flag (1 nếu episode kết thúc, 0 nếu không)

* : Target Q-network parameters (EMA update)

Gradient:

2. Actor Update (Policy Network)

**Objective:** Tối đa hóa expected soft Q-value

Tương đương với minimizing:

Trong đó reparameterization trick:

Sau đó áp dụng squashing function:

**Log-probability với Tanh:**

Trong đó  là Gaussian density trước khi squash.

Gradient:

**3. Temperature Update (Automatic Entropy Tuning)**

Thay vì fix , SAC tự động điều chỉnh temperature parameter để duy trì target entropy.

Objective:

Trong đó  là target entropy, thường được đặt:

(Nghĩa là entropy mong muốn bằng âm của action dimension)

Loss function:

Để đảm bảo , ta tham số hóa:   
Và optimize :

* + 1. Ưu điểm của SAC

1. Sample Efficiency: Off-policy algorithm → có thể reuse data từ replay buffer → cực kỳ sample-efficient

2. Stability:

* Twin Q-networks giảm overestimation
* Entropy regularization tránh premature convergence
* Automatic temperature tuning tự điều chỉnh exploration

3. Stochastic Policy: Policy stochastic tự nhiên → không cần thêm noise (như ε-greedy hay OU noise trong DDPG)

4. Robustness: Hoạt động tốt trên nhiều tasks khác nhau mà không cần tune hyperparameters nhiều

5. Continuous Action Space: Được thiết kế riêng cho continuous control → rất phù hợp với robot manipulation

* + 1. Nhược điểm
* Phức tạp hơn DDPG/TD3 (3 networks cần update)
* Computational cost cao hơn on-policy methods
* Vẫn có thể gặp vấn đề với sparse rewards (→ cần HER/RHER)
  1. Tích hợp SAC với GCRL (SAC-GCRL)
     1. Kiến trúc tổng thể

SAC-GCRL kết hợp sức mạnh của:

* **SAC:** Sample efficiency, stability, automatic exploration
* **GCRL:** Multi-goal learning, generalization

Augmented State: Goal-conditioned Policy: Goal-conditioned Q-function:

A diagram of a system

AI-generated content may be incorrect.

Hình 2.9: SAC-GCRL Agent Architecture

* + 1. Multi-stage Goal Structure

Trong bài toán grasp and lift, chúng tôi phân tách thành 2 stages tuần tự:

A diagram of a diagram

AI-generated content may be incorrect.

Hình 2.10: Multi-stage Goal Structure

Stage 0 - Reach (Tiếp cận):

Goal definition:

(Vị trí của vật thể)

**Success criterion:**

Trong đó  là index của các fingertips,  là threshold (ví dụ: 0.05m).

Reward:

**Stage 1 - Lift (Nhấc lên):**

Goal definition:

Trong đó:

* : Vị trí mục tiêu

* : Hướng mục tiêu (quaternion hoặc rotation matrix)

* : Độ cao tối thiểu (ví dụ: 0. 2m trên mặt bàn)

Success criterion:

Reward:

**Stage Transition:** Chuyển từ Stage 0 → Stage 1 khi:

Multi-stage Policy:

Policy có **multi-head architecture**, mỗi head tương ứng với một stage:

Trong đó  là stage index.

* + 1. Entropy-regularized optimization cho từng stage

Mỗi stage có temperature parameter riêng  để điều chỉnh exploration phù hợp với độ khó.

* **Stage 0 (Reach):** Tương đối dễ → có thể dùng  thấp hơn (ít exploration)
* **Stage 1 (Lift):** Phức tạp hơn → cần  cao hơn (nhiều exploration)

Per-stage temperature update:

Trong đó  là target entropy cho stage .

Target entropy heuristic: Trong đó  là scale factor (ví dụ: ).

Clipping mechanism:

Để tránh  phát triển không kiểm soát:

Ví dụ:

* + 1. Training Procedure

**Thuật toán SAC-GCRL:**

|  |
| --- |
| Initialize:  - Policy network π\_θ with multi-heads  - Q-networks Q\_φ₁, Q\_φ₂ with multi-heads  - Target Q-networks Q\_φ₁', Q\_φ₂'  - Temperature parameters α₀, α₁  - Replay buffer D  - Stage assignment function  For each episode:  1. Sample initial state s₀ and desired goal g^d  2. Determine initial stage j = 0  For t = 0 to T:  3. Select action: a\_t ~ π\_θ(·|o\_t, g^d\_j, j)  4. Execute a\_t, observe r\_t, o\_{t+1}  5. Check stage transition:  if Success\_j(o\_{t+1}, g^d\_j):  j ← j + 1 (advance to next stage)  6. Store transition (o\_t, a\_t, r\_t, o\_{t+1}, g^d\_j, j) in D  For each gradient step:  7. Sample mini-batch B from D  8. Update Critics (per stage):  For each (s, a, r, s', g, j) in B:  y = r + γ(1-d)[min\_i Q\_φᵢ'(s', ã', g, j) - αⱼ log π\_θ(ã'|s', g, j)]  L(φᵢ) = (Q\_φᵢ(s, a, g, j) - y)²  9. Update Actor (per stage):  For each stage j with samples in B:  L(θ) = E[(αⱼ log π\_θ(a|s, g, j) - Q\_φ₁(s, a, g, j))]  10. Update Temperature (per stage): L(αⱼ) = E[-αⱼ(log π\_θ(a|s, g, j) + H̄ⱼ)]  11. Soft update target networks: φᵢ' ← τφᵢ + (1-τ)φᵢ' |

Key implementation details:

1. **Batch composition:** Mỗi batch chứa samples từ cả hai stages, được xử lý riêng biệt.
2. **Stage-specific learning rates:** Có thể sử dụng learning rates khác nhau cho mỗi stage nếu cần.
3. **Success rate tracking:** Monitor success rate của từng stage để điều chỉnh curriculum.
   1. Hindsight Experience Replay (HER)
      1. Vấn đề sparse reward trong robot learning

Trong robot learning, đặc biệt là manipulation tasks, phần thưởng thường rất thưa thớt (sparse):

Vấn đề:

Trong giai đoạn đầu training, robot random explore → xác suất thành công ≈ 0 → hầu hết episodes đều nhận reward = 0.

Với sparse rewards, agent rất khó học vì:

* Không có gradient signal để cải thiện
* Exploration ngẫu nhiên không hiệu quả trong high-dimensional space
* Cần rất nhiều episodes may mắn để thành công lần đầu

Các giải pháp truyền thống:

1. **Reward shaping:** Thiết kế dense rewards

* Nhược điểm: Cần expert knowledge, dễ reward hacking

1. **Curriculum learning:** Bắt đầu với tasks dễ
   * Nhược điểm: Phải thiết kế curriculum thủ công
2. **Demonstration:** Học từ expert demonstrations
   * Nhược điểm: Cần thu thập demonstrations tốn kém
     1. Cơ chế Goal Relabeling

Hindsight Experience Replay (HER) [48] đưa ra một insight đột phá: "Mỗi episode thất bại đều thành công với một goal khác!"

**Ý tưởng:** Sau khi thực hiện một episode, thay vì chỉ lưu trữ với desired goal ban đầu, chúng ta **relabel** (gán lại) transitions với các **hindsight goals** - những goals mà ta đã thực sự đạt được.

Formal definition:

* **Desired goal:**  - Mục tiêu ban đầu
* **Achieved goal:**  - Mục tiêu đạt được tại time
* **Hindsight goal:**  - Mục tiêu được chọn từ tương lai trong cùng episode
  + 1. Các chiến lược sampling

HER đề xuất nhiều strategies để chọn hindsight goals:

1. Future Strategy (Được khuyến nghị nhất)

Chọn  achieved goals từ **future timesteps** trong cùng episode:

**Ưu điểm:**

* Tạo ra nhiều successful transitions
* Goals có liên quan đến trajectory hiện tại
* Curriculum tự nhiên: goals gần hơn dễ học hơn

2. Final Strategy

Luôn chọn achieved goal ở cuối episode:

Ưu điểm:

* Đơn giản
* Mỗi episode tạo ít nhất 1 successful trajectory

Nhược điểm**:** Ít diversity hơn future strategy

3. Episode Strategy

Chọn ngẫu nhiên từ bất kỳ timestep nào trong episode:

4. Random Strategy

Chọn goal ngẫu nhiên từ goal space (không nhất thiết là achieved):

Nhược điểm:

* Có thể chọn goals không realistic
* Hiệu quả thấp
  + 1. Công thức HER Relabeling

Vanilla HER algorithm:

Với xác suất  (thường = 0.8), relabel transition :

**Step 1:** Sample hindsight timestep  **Step 2:** Extract achieved goal at

**Step 3:** Recompute reward với hindsight goal

**Step 4:** Store relabeled transition

|  |
| --- |
| Episode length T = 50  Future strategy với k = 4  For transition at t=10:  Original: (o₁₀, a₁₀, r=0, o₁₁, g^d=[1,2,3])  Sample 4 future timesteps: [15, 23, 38, 50]  Relabeled transitions:  (o₁₀, a₁₀, r'=?, o₁₁, g^h=φ(o₁₅))  (o₁₀, a₁₀, r'=?, o₁₁, g^h=φ(o₂₃))  (o₁₀, a₁₀, r'=?, o₁₁, g^h=φ(o₃₈))  (o₁₀, a₁₀, r'=?, o₁₁, g^h=φ(o₅₀))  Total transitions from this single step: 1 + 4 = 5  Total from episode: 50 \* 5 = 250 transitions! |

Kết quả: Từ 1 episode thất bại, HER tạo ra hàng trăm successful training samples → learning hiệu quả hơn nhiều.

* 1. Relay Hindsight Experience Replay (RHER)
     1. Động lực và thách thức

Mặc dù Hindsight Experience Replay (HER) đã chứng minh hiệu quả vượt trội trong các bài toán Goal-conditioned RL với phần thưởng thưa thớt, phương pháp này vẫn tồn tại những hạn chế đáng kể khi áp dụng vào các nhiệm vụ tuần tự phức tạp (sequential multi-goal tasks).

Vấn đề cốt lõi nằm ở Hiệu ứng Tiêu cực của Việc Gán nhãn lại Phi tuần tự (Negative effect of Non-Sequential Relabeling - NNSR). Trong các nhiệm vụ tuần tự, các mục tiêu phụ (sub-goals) cần được hoàn thành theo một trật tự logic cố định. Tuy nhiên, cơ chế gán nhãn lại của HER không phân biệt giữa các mục tiêu gần và xa, hay giữa các mục tiêu ở các giai đoạn khác nhau. Điều này dẫn đến hiện tượng *Non-Sequential Relabeling*: một mục tiêu của giai đoạn sau (ví dụ: "thả vật thể") có thể được gán ngẫu nhiên làm mục tiêu thay thế cho một bước trong giai đoạn đầu (ví dụ: "di chuyển tới vật thể"). Việc này tạo ra những mẫu huấn luyện không hợp lệ về mặt logic, khiến tác tử học các chiến lược sai lầm và cản trở quá trình hội tụ.

* + 1. Cơ chế Hindsight Experience Replay (HER)

Để hiểu rõ vấn đề, ta cần xem xét cơ chế gốc của HER. Kỹ thuật này được thiết kế để cải thiện hiệu quả học tập trong môi trường phần thưởng thưa thớt. Cơ chế hoạt động dựa trên việc, với một xác suất , mục tiêu mong muốn ban đầu  trong một chuyển trạng thái  sẽ được thay thế bằng một mục tiêu đã đạt được  từ chính episode đó. Bằng cách này, những kinh nghiệm ban đầu bị coi là "thất bại" so với  lại trở thành "thành công" so với , làm tăng mật độ tín hiệu học tập.

Cụ thể, theo phương pháp vanilla-HER, với xác suất , mục tiêu ban đầu  tại bước thời gian  sẽ được gán lại thành một mục tiêu quan sát được (achieved goal) tại một bước thời gian trong tương lai  trong cùng episode, được mô tả bằng công thức:

Kỹ thuật Hindsight Relabeling được áp dụng nhằm cải thiện hiệu quả học tập trong các môi trường có phần thưởng thưa thớt (sparse rewards). Cơ chế hoạt động dựa trên việc, với xác suất , các mục tiêu ban đầu sẽ được thay thế bằng các mục tiêu đã đạt được (achieved goals), từ đó biến những chuyển tiếp được xem là thất bại thành các mẫu dữ liệu thành công. Phương pháp này giúp tăng mật độ tín hiệu học tập, giảm độ khó của bài toán và đặc biệt hiệu quả trong các tình huống mà phần thưởng chỉ xuất hiện khi đạt được mục tiêu cuối cùng.

Theo phương pháp vanilla-HER, với xác suất 0.8, ta sẽ gán lại (relabel) mục tiêu ban đầu bằng một quan sát khác tại một bước thời gian tương lai trong cùng episode, như ở (4):

Trong đó:

* là hindsight goal tại bước ,

* là achieved goal tại bước ,
* là bước thời gian cuối cùng của episode.
  + 1. Kiến trúc RHER

Bốn thành phần chính của RHER phối hợp với nhau để thực hiện quá trình khám phá tự định hướng, giúp tác tử nhanh chóng giải quyết các nhiệm vụ thao tác đối tượng tuần tự phức tạp, như minh họa trong Hình 2.

A diagram of a process

AI-generated content may be incorrect.

Hình 2.11: Sơ đồ của RHER, trong đó các thành phần chính được thể hiện trong các khung màu vàng. Khung này thực hiện khám phá tự định hướng cho một nhiệm vụ tuần tự.

Theo thứ tự logic của thiết kế, bốn thành phần bao gồm:

1. Thành phần đầu tiên là phân tách nhiệm vụ tuần tự ban đầu thành giai đoạn (hoặc bước) và sau đó kết hợp chúng thành nhiệm vụ con với mức độ phức tạp tăng dần;
2. Thành phần thứ hai là thiết kế một mô hình học tăng cường đa mục tiêu & đa nhiệm để xử lý đồng thời các nhiệm vụ con trên;
3. Bằng cách sử dụng HER, thành phần này vượt qua vấn đề phần thưởng thưa thớt tiêu cực (negative sparse rewards) và chia sẻ dữ liệu được thu thập từ các nhiệm vụ con khác nhau;
4. Để thu thập các mẫu có giá trị hơn và giảm nhẹ vấn đề NNSR by HER đối với các nhiệm vụ tuần tự, một Chiến lược Khám phá Tự Định Hướng (Self-Guided Exploration Strategy – SGES) được đề xuất. SGES sử dụng chính sách đã học để dẫn dắt tác tử nhanh chóng đi đến các giai đoạn phức tạp hơn. Thông qua quá trình khám phá tự định hướng này, tác tử có thể học các nhiệm vụ tuần tự từng giai đoạn, tương tự như chạy tiếp sức.

Là một khung học tăng cường liên tục, sự kết hợp của các thành phần này hiện thực hóa quá trình chuyển giao kiến thức ổn định từ một chính sách đơn giản sang một chính sách phức tạp hơn và đảm bảo rằng chính sách đơn giản không bị quên.

* + 1. Thuật toán RHER

|  |
| --- |
| Algorithm 1 Relay Hindsight Experience Replay |
| Khởi tạo tác tử và bộ nhớ replay  **for** *each episode* **do**   Lấy mẫu một observation dictionary   **for** *each time step* **do**    Kiểm tra giai đoạn theo (6)    Lấy mẫu một hành động bằng SGES theo (10)    Thực thi hành động và nhận quan sát mới    Lưu trữ transition   **for** *each gradient step* **do**    **for** *each sub-task* **do**     Lấy mẫu một mini-batch episodes từ      Thay thế mục tiêu theo (9)     Thực hiện tối ưu (1), (2) dựa trên |

A diagram of mathematical equations

AI-generated content may be incorrect.

Hình 2.12: Phân rã và sắp xếp lại các tác vụ theo trình tự.

* + 1. Kỹ thuật huấn luyện

2.9.5.1. N-step Returns

Phương pháp N-step Returns được áp dụng nhằm cải thiện khả năng gán tín dụng theo thời gian trong quá trình huấn luyện. Cụ thể, giá trị hồi quy được tính dựa trên nhiều bước (multi-step learning) với số khung hình , sử dụng hệ số chiết khấu để giảm trọng số của phần thưởng xa trong tương lai. Ngoài ra, một hệ số suy giảm bước (step decay) bằng 0.7 được áp dụng nhằm điều chỉnh mức độ ảnh hưởng của các phần thưởng trung gian, giúp cân bằng giữa tín hiệu ngắn hạn và dài hạn trong việc cập nhật giá trị.

* + - 1. Recurrent Batching

Kỹ thuật Recurrent Batching được áp dụng nhằm duy trì tính liên kết theo thời gian trong quá trình huấn luyện mạng hồi tiếp (SEQGRUNET). Thay vì lấy mẫu các chuyển tiếp đơn lẻ, phương pháp này thực hiện lấy mẫu theo chuỗi (sequence sampling), giúp bảo toàn thông tin phụ thuộc thời gian giữa các bước quan sát. Các chuỗi được tổ chức thành các batch với kích thước 512 mẫu, đảm bảo hiệu quả tính toán đồng thời tối ưu hóa khả năng học của mô hình trong các bài toán có cấu trúc tuần tự.

* + - 1. Chiến lược Priority và Sampling

Chiến lược Priority và Sampling trong hệ thống hiện tại được triển khai dưới dạng lấy mẫu đồng đều (uniform sampling) từ bộ nhớ kinh nghiệm, đảm bảo tính ngẫu nhiên trong việc chọn dữ liệu huấn luyện. Bộ nhớ có dung lượng tối đa 20.000 tập (episodes) nhằm cân bằng giữa khả năng lưu trữ và hiệu quả tính toán. Khi bộ nhớ đạt giới hạn, cơ chế dọn dẹp tự động được kích hoạt để loại bỏ các kinh nghiệm cũ, duy trì tính cập nhật của dữ liệu phục vụ quá trình học.

* 1. Kiến trúc mạng nơ-ron (Neural Network Architecture)

Thiết kế kiến trúc mạng nơ-ron là yếu tố quyết định đến khả năng học và generalization của agent. Nghiên cứu này sử dụng kiến trúc **SeqGRUNet** - một mạng kết hợp recurrent layers và feedforward layers để xử lý sequential observations.

* + 1. Policy Network (SeqGRUNet)

Động lực thiết kế

Trong robot manipulation tasks, quan sát hiện tại có thể không đủ để xác định hành động tối ưu (Partially Observable MDP - POMDP). Ví dụ:

* Velocity estimation: Cần nhiều frames để ước tính vận tốc chính xác
* Contact detection: Xu hướng force/torque qua thời gian quan trọng hơn giá trị tức thời
* Trajectory planning: Cần "nhớ" các hành động trước đó

**Giải pháp:** Sử dụng Recurrent Neural Network (RNN), cụ thể là GRU (Gated Recurrent Unit), để duy trì memory về quá khứ.

Kiến trúc chi tiết

Policy network nhận ba inputs chính:

1. **Observation sequence:**

* + : Sequence length (thường = 10)

* + : Observation dimension (≈ 70)

1. **Desired goal:**

* : Goal dimension (tùy stage)
* Được concatenate với mỗi observation trong sequence

1. **Attention mask:**
   * Mask out padding positions
   * 1 = valid timestep, 0 = padding

Bảng 2.2: Các thành phần kiến trúc mô hình

| Component | Lý do sử dụng |
| --- | --- |
| GRU | Nhẹ hơn LSTM, hiệu quả với sequences ngắn (10 steps) |
| Single GRU layer | Tránh overfitting, đủ capacity cho task |
| SiLU activation | Smooth, non-monotonic → better gradients than ReLU |
| Layer normalization | Stabilize training, faster convergence |
| Multi-head output | Riêng biệt cho mỗi stage → tránh interference |
| Orthogonal init | Preserve gradient magnitude qua layers |

* + 1. Value Network (Critic)

Value network ước tính Q-value cho cặp (state, action, goal) để hướng dẫn policy learning.

Mạng giá trị (Value Network hay Critic) được thiết kế với cấu trúc tương tự Policy Network, nhưng đầu vào bao gồm chuỗi quan sát mặt nạ để xử lý chuỗi biến đổi, cùng với thông tin meta được ghép từ mục tiêu và hành động trúc mạng sử dụng một lớp GRU (embed\_dim = 256, num\_layers = 1, có chuẩn hóa) kết hợp với các lớp MLP [1024, 768, 512] với hàm kích hoạt SiLU, tương tự như Policy Network. Đầu ra ban đầu có dạng , nhưng sau khi chọn theo task\_id được reshape về để phục vụ tính toán Q-value cho tác vụ hiện tại. Trong quá trình huấn luyện, thuật toán Soft Actor-Critic (SAC) được triển khai với các cơ chế quan trọng:

* + 1. Sử dụng hai Q-head song song (Twin Q-networks) thông qua EnsembleDeepValueFunc để giảm hiện tượng đánh giá quá cao
    2. Áp dụng mạng mục tiêu với kỹ thuật Polyak averaging () nhằm ổn định cập nhật
    3. Bổ sung thành phần entropy với hệ số nhiệt độ có thể học được, được tối ưu hóa thông qua reg\_ent\_losses để cân bằng giữa khai thác và khám phá.

Value network nhận **bốn** inputs:

1. **Observation sequence:**
2. **Attention mask:**
3. **Meta information:**
   * Goal và action được concatenate
   * Inject vào MLP layers (không qua GRU)
     1. Target Networks

Để ổn định quá trình học tập trong SAC, hai bộ target networks được duy trì song song với main networks.

Trong thuật toán off-policy, việc sử dụng cùng một network để tính cả current estimate và target value có thể dẫn đến moving target problem - mục tiêu liên tục thay đổi trong quá trình học, gây mất ổn định.

Bellman target với target network:

Trong đó:

* : Tham số của target Q-network thứ

* : Tham số của target policy network

* : Action được sample từ current policy (không phải target policy)

Polyak Averaging Update

Thay vì copy định kỳ như trong DQN, SAC sử dụng **soft update** (Polyak averaging):

Trong đó  là update rate, thường rất nhỏ ().

Ý nghĩa:

* nhỏ → target networks thay đổi chậm → ổn định cao
* lớn → target networks cập nhật nhanh → phản ánh policy mới nhanh hơn

Phân tích toán học:

Tại iteration , giá trị target parameter là:

Đây là exponential moving average (EMA) với decay factor .

Với :

* Sau 100 updates: weight của
* Sau 500 updates: weight của

Lợi ích:

1. Smooth target evolution: Targets không nhảy đột ngột
2. Reduced variance: Averaging effect giảm noise
3. Improved stability: Tránh oscillations trong training
   * 1. Temperature và Penalty Coefficients

Learnable Temperature Parameters

Mỗi stage có temperature parameter riêng  để điều chỉnh mức độ exploration phù hợp.

Tham số hóa:

Để đảm bảo , ta sử dụng log-parameterization:

Và optimize  thay vì  trực tiếp.

Objective function:

Gradient:

**Dual optimization interpretation**:

Bài toán tối ưu entropy-augmented objective có thể được viết dưới dạng constrained optimization:

Sử dụng Lagrange multiplier  (temperature), ta có Lagrangian:

Tối ưu hóa xen kẽ:

* **Primal:** Tối đa hóa theo  (actor update)
* **Dual:** Tối thiểu hóa theo  (temperature update)

Clipping Mechanism

Để tránh  phát triển không kiểm soát được, áp dụng clipping:

Trong nghiên cứu này:

* : Đảm bảo luôn có một mức exploration tối thiểu

* : Tránh exploration quá mức làm giảm exploitation

Gradient clipping:

Khi  chạm boundary, gradient được project:

* 1. Tổng quan hệ thống nghiên cứu

Hệ thống được thiết kế với kiến trúc modular, bao gồm các thành phần chính:

1. Environment Layer: Môi trường mô phỏng Isaac Lab với Franka-Shadow Hand robot
2. Agent Layer: SAC agent với policy và value networks dựa trên SeqGRUNet
3. Memory Layer: RHER memory buffer để lưu trữ và sample experience
4. Learning Layer: RHER learner quản lý vòng lặp huấn luyện

Quy trình hoạt động của hệ thống được triển khai như sau: trước hết, agent tương tác trực tiếp với môi trường để thu thập dữ liệu kinh nghiệm theo chính sách hiện tại. Các kinh nghiệm này sau đó được lưu trữ trong bộ nhớ RHER, kết hợp với kỹ thuật hindsight relabeling và stage mixing (với tỷ lệ r\_mix=0.5) nhằm tăng hiệu quả học tập trong môi trường phần thưởng thưa thớt. Tiếp theo, agent sẽ học từ các batch kinh nghiệm được lấy mẫu ngẫu nhiên từ bộ nhớ này, cập nhật actor, critic, và entropy coefficient thông qua thuật toán SAC.

A diagram of a environment

AI-generated content may be incorrect.

Hình 2.13: Tổng quan kiến trúc mô hình

Trong nghiên cứu này, nhiều thuật toán điều khiển liên tục phổ biến như PPO, SAC, DDPG, SAC+HER và DDPG+HER đã được xem xét. PPO có tính ổn định cao nhưng hiệu quả mẫu thấp trong môi trường robot phức tạp. DDPG và DDPG+HER đạt hiệu suất tốt với hành động liên tục nhưng dễ mất ổn định và nhạy với nhiễu. SAC và SAC+HER có sample efficiency vượt trội nhưng không xử lý tốt phụ thuộc chuỗi thời gian và multi-goal learning. Dựa trên các hạn chế này, nghiên cứu đề xuất SAC kết hợp RHER và chính sách SEQGRUNET nhằm tối ưu hóa khả năng khám phá, cải thiện ổn định và xử lý hiệu quả chuỗi quan sát theo thời gian.

1. KẾT QUẢ THỰC NGHIỆM
   1. Cấu hình môi trường
      1. Tổng quan về Isaac Sim

Isaac Sim là nền tảng mô phỏng robot của NVIDIA, xây dựng trên Universal Scene Description (USD) và sử dụng bộ máy vật lý PhysX 5 với khả năng tăng tốc bằng GPU. Isaac Sim hỗ trợ đầy đủ các thành phần quan trọng bao gồm mô phỏng động lực học, mô phỏng cảm biến, chiếu sáng, dựng hình (rendering) thời gian thực và khả năng mở rộng để tích hợp trực tiếp với các thư viện học tăng cường sâu thông qua Isaac Lab. Với kiến trúc tối ưu cho GPU, Isaac Sim mang lại tốc độ mô phỏng cao hơn đáng kể so với các engine truyền thống, giúp tăng tốc quá trình thu thập dữ liệu kinh nghiệm trong RL và hỗ trợ huấn luyện mô hình hiệu quả hơn.

* + 1. Cấu hình môi trường mô phỏng

Môi trường mô phỏng được thiết kế dựa trên kiến trúc Manager-Based RL Environment của Isaac Lab. Mỗi môi trường (ENV) được cấu hình với:

* Số lượng môi trường song song: 8 ENV
* Khoảng cách giữa các ENV: 2.5 m
* Bộ mô phỏng vật lý: PhysX 5 (GPU), với các tham số tối ưu hóa va chạm, số vòng lặp solver và buffer GPU
* Thời gian bước mô phỏng: dt = 0.025 s
* Độ dài mỗi tập (episode): 3.2 giây
* Tần suất lựa chọn hành động: được điều chỉnh bởi decimation = 2
  + 1. Các đối tượng được mô phỏng

Bảng 3.1: Các thành phần cố định của môi trường

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Thành phần** | **File mô hình / Cấu hình** | **Vị trí ban đầu** | **Tư thế / Thuộc tính** | **Đặc trưng** |
| Robot Franka FR3 + Shadow Hand | Franka\_robot.usd | – | 31 DoF (FR3: 7, Shadow Hand: 24) | Thực hiện thao tác tiếp xúc phức tạp, phạm vi chuyển động rộng |
| Bàn thao tác (Seattle Lab Table) | table\_instanceable.usd | (0.5, 0.0, 0.0) | Xoay 90° quanh trục X | Bề mặt ổn định để đặt vật; sát không gian thực |
| Mặt đất (Ground Plane) | GroundPlaneCfg | (0.0, 0.0, –1.05) | – | Mặt phẳng vật lý chuẩn, ngăn vật thể rơi ra ngoài |
| Hệ thống chiếu sáng (Dome Light) | DomeLight | – | Màu: (0.75, 0.75, 0.75) |  |

Bảng 3.2: Các vật thể dùng cho thao tác

|  |  |  |  |
| --- | --- | --- | --- |
| **Vật thể** | **DexCube** | **Mustard Bottle (YCB)** | **Drill (YCB)** |
| **File mô hình** | dex\_cube\_instanceable.usd | mustard\_bottle.usd | drill.usd |
| **Tỉ lệ (scale)** | (1.2, 1.2, 1.2) | (0.6, 0.6, 0.6) | (1.0, 1.0, 1.0) |
| **Khối lượng** | Density = 400 | 0.44 kg | 1.2 kg |
| **Vị trí ban đầu** | (0.3, 0.0, 0.08) | (0.3, 0.0, 0.05) | (0.3, 0.0, 0.05) |
| **Tư thế ban đầu** | Hướng mặc định | Xoay 90° quanh trục X | Hướng mặc định |
| **Đặc trưng** | Hình lập phương đơn giản, phù hợp cho học hành vi cơ bản | Dạng chai trụ, khó gắp hơn khối lập phương, dùng để đánh giá khả năng khái quát hóa. | Vật thể dạng dài, trọng tâm lệch, phù hợp cho các bài toán ổn định và lập kế hoạch gắp. |

A robot arm on a black box

AI-generated content may be incorrect.

Hình 2.14: Thiết lập môi trường thực nghiệm

A white robot legs on a black box

AI-generated content may be incorrect.

Hình 2.15: Cánh tay robot thực hiện tác vụ

A robotic arm on a black box

AI-generated content may be incorrect.

Hình 2.16: Cánh tay robot hoàn thành tác vụ

* + 1. Mô phỏng tiếp xúc và vật lý

Một trong các điểm mạnh lớn nhất của Isaac Sim là khả năng mô phỏng tiếp xúc chân thực nhờ PhysX 5. Trong nghiên cứu này, các cấu hình quan trọng được bật gồm:

* **enable\_gyroscopic\_forces:** Giúp mô phỏng chuyển động quay ổn định hơn khi vật thể bị xoay nhanh, tránh rung hoặc mất ổn định.
* **solver\_position\_iteration\_count = 8**:Tăng số vòng lặp giải pháp tiếp xúc, giúp mô phỏng chính xác hơn tại điểm ngón tay chạm vào vật thể (đặc biệt quan trọng với Shadow Hand).
* **max\_depenetration\_velocity = 1000**:Giảm hiện tượng “xuyên vật” bằng cách buộc vật thể thoát khỏi trạng thái chồng lấp nhanh hơn.
* **sleep\_threshold = 0.005**:Giúp vật thể không bị “rơi vào trạng thái ngủ” khi robot tương tác nhẹ, duy trì độ nhạy phản hồi trong suốt quá trình gắp.
* **stabilization\_threshold = 0.0025**:Ổn định va chạm nhỏ giữa các bề mặt, giảm rung khi tiếp xúc nhiều điểm.

Đối với tác vụ gắp bằng Shadow Hand – vốn có nhiều điểm tiếp xúc nhỏ và phức tạp – các tham số của PhysX đóng vai trò quan trọng để đảm bảo bàn tay robot phản hồi chính xác khi tương tác với vật thể. Điều này cho phép robot học cách điều khiển lực, vị trí ngón tay và tạo ra thế nắm ổn định, một điều rất khó đạt được khi sử dụng engine có mô hình tiếp xúc đơn giản

* + 1. Tích hợp mô phỏng với hệ thống học tăng cường sâu

Isaac Lab cung cấp một framework API giúp kết nối môi trường mô phỏng với agent RL một cách trực tiếp. Trong nghiên cứu này:

Mỗi ENV cung cấp observation gồm:

* Trạng thái robot (joint position, velocity)
* Vị trí và hướng của vật thể (achieved\_goal)
* Mục tiêu đặt ra (desired\_goal)
* Trạng thái các đầu ngón tay, khung bàn tay

**Action space:** 30 chiều (6 lệnh IK + 24 khớp ngón tay)

**Observation space:** 134 chiều

**Ưu điểm của việc tích hợp Isaac Lab:**

* Hỗ trợ multi-environment chạy song song, tăng tốc thu thập dữ liệu
* Giao diện RL được chuẩn hóa, dễ đồng bộ với SAC, HER hoặc PPO
* Đảm bảo độ ổn định và tái lập (reproducibility) trong quá trình training
  + 1. So sánh với các nền tảng mô phỏng khác

Bảng 2.3: So sánh Isaacsim, Mujoco và Pybullet

|  |  |  |  |
| --- | --- | --- | --- |
| **Tiêu chí** | **IsaacSim (NVIDIA)** | **Mujoco** | **PyBullet** |
| **Engine vật lý** | PhysX 5 (GPU + CPU) | MuJoCo Physics (CPU) | Bullet Physics (CPU) |
| **Độ chính xác tiếp xúc (contact fidelity)** | Rất cao – mô phỏng tiếp xúc phức tạp, phù hợp cho Shadow Hand | Cao nhưng kém chính xác hơn IsaacSim trong bài toán nhiều điểm tiếp xúc | Trung bình, contact còn đơn giản |
| **Hỗ trợ GPU** | Có – tăng tốc mô phỏng multi-env | Không | Không |
| **Tốc độ mô phỏng nhiều môi trường** | Rất tốt nhờ GPU PhysX | Tốt nhưng bị giới hạn CPU | Tốt nhưng dễ không ổn định khi scale lớn |
| **Rendering / USD / chất lượng hình ảnh** | Rất cao, dùng RTX, USD pipeline | Cơ bản, không tập trung vào rendering | Trung bình, đơn giản |
| **Thư viện mô hình / Asset chất lượng** | Phong phú, hỗ trợ USD, robot thật (Franka, UR5, Shadow Hand) | Tốt, tài sản chuẩn cho research | Tương đối, nhưng không phong phú bằng |
| **Mô phỏng cảm biến (camera, depth, tactile)** | Mạnh nhất, hỗ trợ RTX Sensor, camera thực tế | Cơ bản | Có nhưng chất lượng thấp |
| **Dễ dùng** | Trung bình – cần GPU mạnh và kiến thức USD | Dễ dùng, gọn nhẹ | Dễ dùng nhất, API đơn giản |
| **Tích hợp RL** | Rất tốt (Isaac Lab, multi-env, vectorized env) | Tốt (một số wrapper chuẩn) | Tốt (nhiều ví dụ, dễ tích hợp) |
| **Phù hợp cho mô hình phức tạp hoặc bài toán contact-rich** | Tốt | Tốt | Không phù hợp |
| **Sim-to-Real** | Rất mạnh, vì có RTX + USD + chính xác vật lý cao | Trung bình | Thấp |
| **Cài đặt / yêu cầu phần cứng** | Nặng – cần GPU mạnh (RTX) | Nhẹ | Rất nhẹ |
| **Mức độ ổn định** | Cao, đặc biệt với môi trường phức tạp | Rất cao | Trung bình |

* 1. Học tăng cường sâu (DRL)
     1. Môi trường thực nghiệm

Trong mô phỏng môi trường, chúng ta sử dụng một cánh tay robot Franka Emika FR3 với 7 bậc tự do, được nối với bàn tay Shadow Hand. Mô phỏng này sử dụng động cơ vật lý Pybullet, cho phép cánh tay robot Franka Emika FR3 cùng với Shadow Hand song song thao tác với bốn vật thể khác nhau: **khối lập phương, máy khoan, chai tương ớt.**

* + 1. Không gian trạng thái (Observation Space)

Không gian quan sát được thiết kế để cung cấp đầy đủ thông tin về trạng thái hiện tại của robot và vật thể, cho phép agent đưa ra quyết định tối ưu. Observation vector tại timestep  được định nghĩa:

Bảng 3.1: Định nghĩa các thành phần trong mỗi trạng thái

| **Thành phần** | **Ký hiệu** | **Dimension** | **Mô tả** |
| --- | --- | --- | --- |
| **Vị trí vật thể** |  | 3 | Tọa độ (x, y, z) trong robot base frame |
| **Hướng vật thể** |  | 3 | Euler angles (roll, pitch, yaw) |
| **Vị trí end-effector** |  | 3 | Tọa độ palm center |
| **Hướng end-effector** |  | 3 | Euler angles của bàn tay |
| **Vị trí fingertips** |  | 15 | Vị trí 3D của 5 fingertips |
| **Hướng fingertips** |  | 15 | Euler angles của 5 fingertips |
| **Góc khớp cánh tay** |  | 7 | Joint positions của Franka FR3 |
| **Góc khớp bàn tay** |  | 24 | Joint positions của Shadow Hand |
| **Vận tốc khớp cánh tay** |  | 7 | Joint velocities của FR3 |
| **Vận tốc khớp bàn tay** |  | 24 | Joint velocities của Shadow Hand |
| **Hành động trước** |  | 25 | Previous action (proprioception) |
| **Padding** | - | 5 | Zero padding để đạt dimension chuẩn |
| **Tổng** |  | **134** | Complete observation vector |

**Lưu ý về thiết kế:**

1. **Proprioceptive information:** Góc khớp và vận tốc cung cấp thông tin về cấu hình hiện tại của robot
2. **Exteroceptive information:** Vị trí/hướng vật thể và fingertips cho phép robot nhận thức môi trường
3. **History information:**  giúp policy học các behaviors có tính tuần tự (sequential)
4. **Normalization:** Tất cả components được chuẩn hóa về range  trước khi đưa vào network
   * 1. Không gian hành động (Action Space)

Không gian hành động được định nghĩa là không gian liên tục (continuous) với dimension tương ứng với số bậc tự do điều khiển được:

**Cấu trúc action vector:**

Trong đó:

* : Target joint positions cho Franka FR3
* : Target joint positions cho Shadow Hand

**Chế độ điều khiển:**

Hệ thống sử dụng **position control mode** với PD controller ở mức thấp:

Trong đó:

* : Torque áp dụng lên khớp
* : Proportional và derivative gains
* : Target position từ policy
* : Current position và velocity

**Action scaling:**

Actions từ policy network (range ) được scale về joint limits thực tế:

* + 1. Multi-stage Goal Encoding

Bài toán grasp-and-lift được phân rã thành hai stages tuần tự với goal encoding riêng biệt.

**Stage 0 - Reach (Tiếp cận vật thể):**

Mục tiêu: Di chuyển bàn tay đến gần vật thể.

**Desired goal:**

**Achieved goal:**

**Success criterion:**

với  m.

**Stage 1 - Lift (Nắm và nhấc):**

Mục tiêu: Nắm vật thể và nhấc lên độ cao mong muốn.

**Desired goal:**

Trong đó:

* : Vị trí mục tiêu (x, y, z)
* m: Độ cao tối thiểu so với mặt bàn

**Achieved goal:**

**Success criterion:**

Trong đó:

với  m.

Bảng 3.2: Tổng hợp Goal Encoding cho Multi-stage Task

| **Stage** | **Goal Type** | **Dimension** | **Components** | **Threshold** |
| --- | --- | --- | --- | --- |
| 0 | Desired |  |  | m |
| 0 | Achieved |  |  | - |
| 1 | Desired |  |  | m |
| 1 | Achieved |  |  | m |

**Giải thích lựa chọn encoding:**

Khác với các nghiên cứu multi-object manipulation [49] sử dụng goal encoding phức tạp dạng tensor , bài toán single-object của chúng tôi cho phép sử dụng **simple vector encoding** với:

* **Ưu điểm:**
  + Đơn giản, dễ implement và debug
  + Giảm dimension của input space → faster training
  + Đủ thông tin cho bài toán cụ thể
* **Nhược điểm:**
  + Không trực tiếp encode object type information
  + Policy phải tự học phân biệt objects từ observation
    1. Hàm phần thưởng (Reward Function)

Nghiên cứu sử dụng **sparse reward** để tránh phụ thuộc domain knowledge và khuyến khích agent tự khám phá chiến lược tối ưu.

**Reward cho Stage 0:**

**Reward cho Stage 1:**

* + 1. Điều kiện kết thúc Episode (Termination)

Episode kết thúc khi một trong các điều kiện sau được thỏa mãn:

1. **Timeout:**  timesteps
2. **Task success:**  (hoàn thành cả hai stages)
3. **Object dropped:**  m (vật thể rơi xuống bàn)
4. **Out of workspace:**
5. **Robot limit violation:** Bất kỳ khớp nào vượt quá safety limits
   * 1. Domain Randomization

Để tăng khả năng generalization, các yếu tố sau được randomize ở mỗi episode:

* + - 1. Object Randomization

**Lựa chọn vật thể (Object Selection):**

Tại mỗi episode reset, một trong ba loại vật thể được chọn ngẫu nhiên với xác suất đồng đều:

OBJECT\_TYPES = {cube, mustard\_bottle, drill}

**Đặc tính các vật thể:**

Bảng 3.3: Thông số vật lý của các vật thể

| **Vật thể** | **Kích thước** | **Khối lượng** | **Vị trí khởi tạo (x, y, z)** |
| --- | --- | --- | --- |
| Cube (DexCube) | 0.04 × 0.04 × 0.04 m | Density = 400 kg/m³ | (0.3, 0.0, 0.03) m |
| Mustard Bottle | Cao 0.12 m, ∅ 0.04 m | 0.44 kg | (0.3, 0.0, 0.05) m |
| Drill | Dài 0.15 m, ∅ 0.03 m | 1.2 kg | (0.3, 0.0, 0.05) m |

**Cơ chế hoạt động:**

Cả ba vật thể đều được spawn trong scene, nhưng tại mỗi environment và mỗi episode:

* Chỉ một object được set visible và active (physics enabled)
* Hai objects còn lại được set invisible và moved underground (z = -10m) để tránh tương tác vật lý

Cơ chế này cho phép switching nhanh giữa các objects mà không cần destroy/create objects mới, tối ưu hóa performance.

* + - 1. Initial Pose Randomization

**Trong task variant "lift" (standard):**

Vị trí khởi tạo của vật thể là **cố định** theo Bảng 3.11 ở trên. Orientation khởi tạo cũng cố định:

= (1. 0, 0.0, 0.0, 0.0) [quaternion identity - không rotation]

**Trong task variant "lift\_orientation":**

Vị trí và hướng được randomize:

Vị trí:

* x ~ U([-0.2, 0.2]) m
* y ~ U([-0.2, 0.2]) m
* z: Giữ nguyên theo object type

Hướng (Euler angles):

* roll ~ U([-0.3, 0.3]) rad (≈ ±17°)
* pitch ~ U([-0.3, 0.3]) rad (≈ ±17°)
* yaw ~ U([-π, π]) rad (full rotation)

Các góc Euler sau đó được convert sang quaternion để sử dụng trong simulation.

* + - 1. Goal Position Randomization

Target position cho Stage 1 (Lift) được sample từ uniform distribution với các ranges sau:

x\_target ~ U([0.2, 0.4]) m y\_target ~ U([-0.15, 0. 15]) m z\_target ~ U([0.2, 0.35]) m

Bảng 3.4: So sánh Object Initial Position và Goal Ranges

| **Dimension** | **Object Init (center)** | **Goal Range** | **Range Width** |
| --- | --- | --- | --- |
| x | 0.3 m | [0.2, 0.4] m | 0.2 m |
| y | 0.0 m | [-0.15, 0.15] m | 0.3 m |
| z | 0.03-0.05 m | [0.2, 0.35] m | 0.15 m |

Goal position được randomize trong workspace bounds để đảm bảo:

1. **Reachability:** Tất cả goal positions đều nằm trong workspace của robot
2. **Vertical constraint:** z\_target luôn cao hơn object init → khuyến khích lifting behavior
3. **Lateral diversity:** x, y có range đủ rộng để policy học generalize
   1. Quy trình huấn luyện

Quy trình huấn luyện của mô hình RHER Learner được tổ chức theo cấu trúc vòng lặp nhiều giai đoạn với các siêu tham số được thiết lập nhằm tối ưu hóa hiệu quả học. Cụ thể, quá trình huấn luyện diễn ra trong 2000 epoch, mỗi epoch bao gồm 200 chu kỳ (có thể cấu hình), với 8 môi trường chạy song song để tăng tốc độ thu thập dữ liệu. Trong mỗi chu kỳ, mô hình thực hiện 128 lần cập nhật (configurable), sử dụng batch có kích thước 512 mẫu. Thuật toán tối ưu hóa được triển khai với AdamW, tốc độ học đặt ở mức . Ngoài ra, hệ số chiết khấu được áp dụng để cân bằng giữa phần thưởng ngắn hạn và dài hạn, trong khi xác suất hindsight relabeling được thiết lập ở mức 0.8 nhằm tăng hiệu quả học trong môi trường có phần thưởng thưa thớt.

Quy trình huấn luyện được triển khai theo các bước sau: trước hết, hệ thống thu thập các tập dữ liệu (episodes) từ nhiều môi trường chạy song song. Các tập này được lưu trữ trong bộ nhớ RHER, kết hợp với kỹ thuật hindsight relabeling để tăng hiệu quả học tập trong môi trường có phần thưởng thưa thớt. Tiếp theo, các batch dữ liệu được lấy mẫu từ bộ nhớ và sử dụng để thực hiện cập nhật gradient cho cả mạng Actor và Critic. Sau mỗi lần cập nhật, các mạng mục tiêu (target networks) được điều chỉnh nhằm đảm bảo tính ổn định của quá trình học. Hệ thống được đánh giá định kỳ sau mỗi 50 tập và lưu trữ các điểm kiểm tra (checkpoints) để phục vụ việc khôi phục hoặc phân tích sau này.

Quá trình tối ưu hóa được triển khai với một số kỹ thuật quan trọng nhằm đảm bảo hiệu quả và tính ổn định trong huấn luyện. Thuật toán sử dụng bộ tối ưu hóa AdamW với hệ số weight decay bằng 0.0 để tránh hiện tượng suy giảm trọng số không mong muốn. Các mạng mục tiêu được cập nhật thông qua kỹ thuật Polyak averaging với hệ số , giúp giảm dao động trong quá trình học. Ngoài ra, cơ chế gradient clipping được áp dụng khi cần thiết nhằm kiểm soát độ lớn của gradient, hạn chế hiện tượng bùng nổ gradient. Cuối cùng, huấn luyện được thực hiện với mixed precision training trên GPU, cho phép tăng tốc độ tính toán và giảm yêu cầu bộ nhớ mà vẫn duy trì độ chính xác cần thiết.

Bảng 3.5: So sánh đặc điểm và hiệu quả của các thuật toán

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Thuật toán** | **On/Off Policy** | **Loại chính sách** | **Hiệu quả mẫu** | **Ổn định** | **Phù hợp robot manipulation** |
| PPO | On-policy | Stochastic | Thấp | Cao | Trung bình |
| DDPG | Off-policy | Deterministic | Trung bình | Thấp | Trung bình |
| SAC | Off-policy | Stochastic | Cao | Cao | Rất tốt |
| DDPG+RHER | Off-policy | Deterministic | Tốt | Trung bình | Tốt |
| DDPG+HER | Off-policy | Deterministic | Tốt | Trung bình | Tốt |
| SAC+RHER (proposed) | Off-policy | Stochastic | Rất cao | Rất cao | Xuất sắc |
| SAC+HER | Off-policy | Recurrent | Rất cao | Cao | Rất tốt |

Mô hình được huấn luyện với tối đa 2000 epoch cho tác vụ gắp

**Cấu trúc huấn luyện:**

* Mỗi epoch gồm 200 chu kỳ
* Mỗi chu kỳ gồm 64 bước thời gian
* Chính sách của agent được cập nhật 128 lần mỗi chu kỳ
* Mini batch có kích thước 512 mẫu

**Đánh giá và khám phá:**

* 50-episode đánh giá được thực hiện vào cuối mỗi chu kỳ
* Tỷ lệ trộn r\_mix = 0.5 cho multi-stage guidance

**Bộ đệm phát lại và kỹ thuật HER:**

* Bộ đệm phát lại có dung lượng tối đa 20.000 episode
* HER với để tạo ra các mục tiêu bổ sung
* RHER với tỷ lệ trộn = 0.5 để cải thiện hiệu quả mẫu

**Các tham số khác:**

* Hệ số chiết khấu (γ): 0.98
* Phần thưởng được chia tỷ lệ với hệ số 1.0
* Giá trị trả về được giới hạn ở mức tối đa không giới hạn

**Cấu hình thuật toán:**

**SAC:**

* + Ensemble: 2 Q-networks
  + Độ không chắc chắn (β): Sử dụng minimum Q-value từ ensemble
  + Số lượng hệ số entropy: = số lượng giai đoạn (J)
  + HER-SAC: 1 coefficient
  + RHER-SAC: 2 coefficients
  + Hệ số entropy khởi tạo (α₀): 1.0, phạm vi (0.0, 1.0)
  + Target entropy: 1.0 × action\_dim = 30.0
  + Log std range: (-6.0, 2.0)
  + Tự động điều chỉnh entropy: Có
  + Polyak averaging (τ): 0.005
  + Loss function: Mean Squared Error (MSE)

**DDPG:**

* + Tỷ lệ hành động ngẫu nhiên: 0.05
  + Nhiễu hành động (action noise): 0.1
  + Q-networks: 1 (single critic)
  + Target networks: Có (actor + critic)
  + Polyak averaging (τ): 0.005
  + Loss function: Mean Squared Error (MSE)
  + Policy type: Deterministic
  + Action space: Continuous (Box)

**PPO:**

* + Clip ratio (ε): 0.2
  + Value clip ratio: 0.2
  + Entropy coefficient: 0.01
  + Value loss coefficient: 0.5
  + GAE lambda (λ): 0.95
  + Log std range: (-6.0, 2.0)
  + Policy type: Stochastic
  + Value function: V(s) - State value
  + No replay buffer: On-policy learning

**Tối ưu hóa và kiến trúc mạng:**

* Bộ tối ưu hóa AdamW
* Learning rate: 3e-4 (0.0003)
* Polyak averaging: 0.005
* Betas: (0.9, 0.999)
* Weight decay: 0.0
* Epsilon: 1e-8

Hình 16 dưới đây minh họa kiến trúc chi tiết của các mạng nơ-ron được sử dụng trong quá trình huấn luyện các thuật toán Deep Reinforcement Learning (DRL) cho tác vụ điều khiển robot. Kiến trúc này bao gồm các thành phần chính như mạng Actor, mạng Critic, và các mạng phụ trợ khác, được thiết kế để xử lý thông tin đầu vào (như hình ảnh, trạng thái robot) và đưa ra quyết định hành động tối ưu trong môi trường mô phỏng.

A screenshot of a diagram

AI-generated content may be incorrect.

Hình 3.1: Kiến trúc các mạng neural networks được sử dụng cho các thuật toán DRL

Để đánh giá mức độ chính xác và khả năng hoàn thành tác vụ của mô hình điều khiển cánh tay robot, nghiên cứu sử dụng hai chỉ số đánh giá chính:

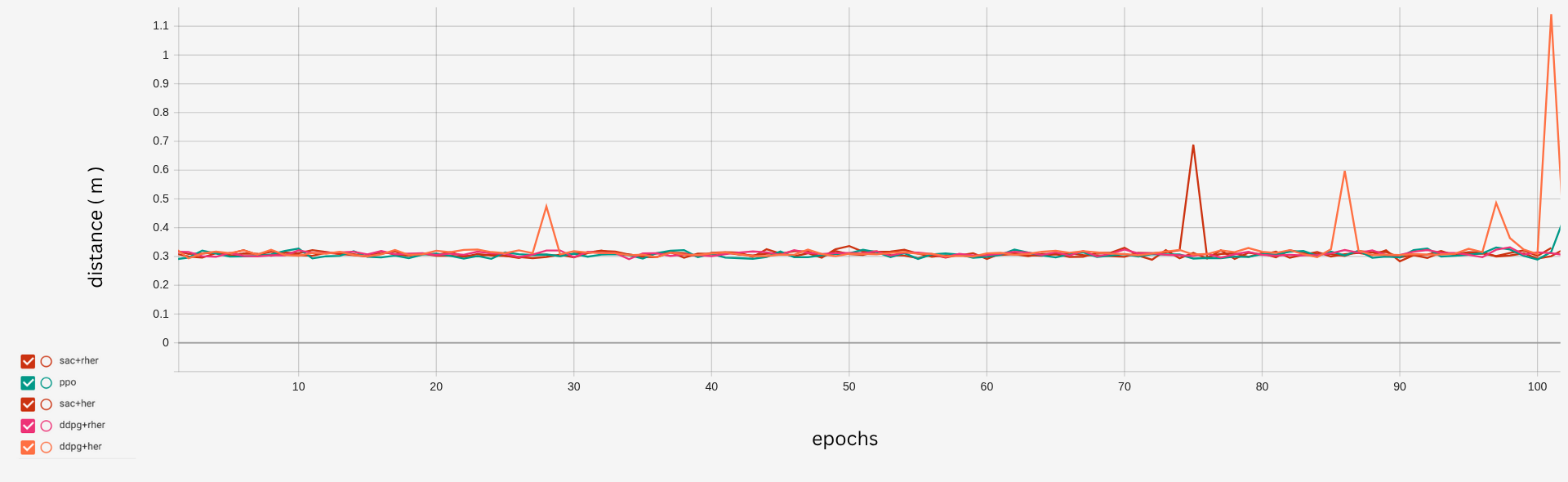
* **Distance**: Là khoảng cách giữa vị trí hiện tại của vật thể và vị trí mục tiêu.
* **Goal\_Achieved**: Là tỷ lệ mức độ hoàn thành mục tiêu, với giá trị tiến gần đến 1 biểu thị khả năng đạt mục tiêu cao.

Kết quả được tóm tắt trong Bảng 3.5 và Hình 3.2, 3.3.

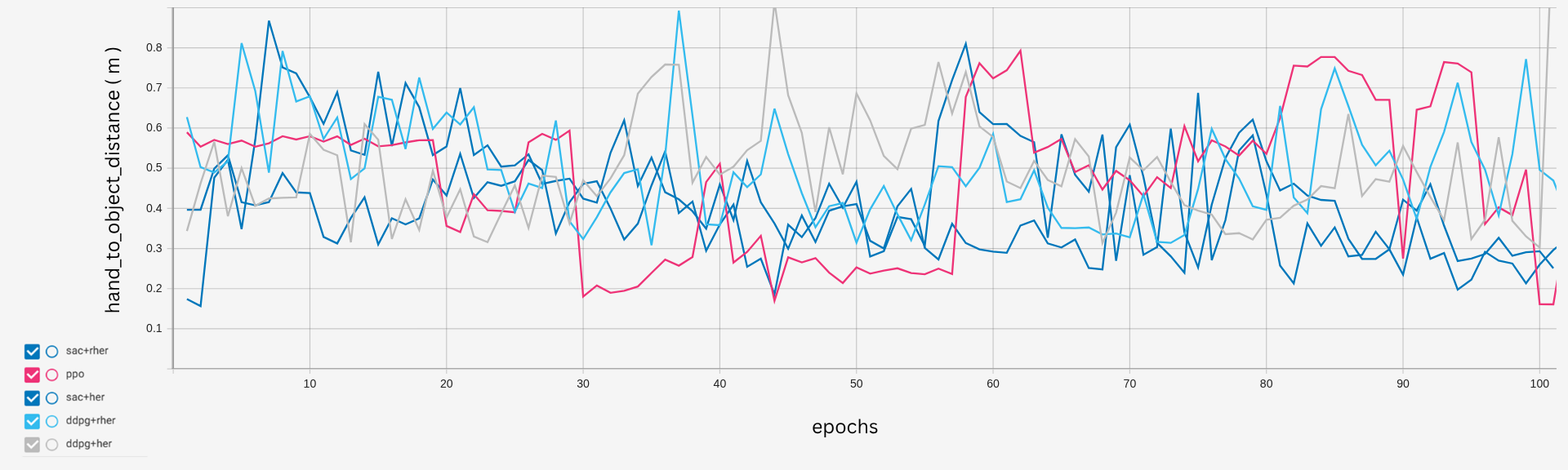
Do các phương pháp vẫn đang trong trạng thái train chưa xong nên các giá trị trong bảng và biểu đồ chúng em sẽ cập nhật chính xác ngay khi chạy xong ạ.

Bảng 3.6: So sánh hiệu năng của các phương pháp trên tác vụ gắp

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Methods** | **Distance** | **Goal\_achieved** | **Time to reach** | **Epoch** |
| SAC + RHER | 0.29 | 1 | 5d20h47m | 687 |
| SAC + HER | 0.18 | 1 | 5d24h0m | 722 |
| DDPG + HER | 0.405 | 1 | 6d3h12m | 1024 |
| DDPG + RHER | 0.409 | 1 | 6d0h12m | 980 |
| PPO | 0.24 | 1 | inf | 2000 |



Hình 3.2: Hiệu năng của các phương pháp trên tác vụ gắp (Distance)



Hình 3.3: Hiệu năng của các phương pháp trên tác vụ gắp (Goal\_achived)

**Phân tích kết quả:**

* **Các thuật toán dựa trên SAC thể hiện hiệu năng nổi trội trong tác vụ gắp khi đánh giá theo hai chỉ số Distance và Goal\_Achieved, với độ ổn định cao hơn đáng kể so với PPO và DDPG. Trong số đó, SAC + RHER cho kết quả tốt nhất: giá trị Distance duy trì ở mức thấp, biên độ dao động nhỏ và không xuất hiện các đỉnh nhiễu lớn. Điều này thể hiện khả năng tiếp cận vật thể ổn định cũng như quỹ đạo điều khiển mượt mà. Cơ chế RHER giúp tận dụng lại các trải nghiệm chưa thành công một cách hiệu quả, đặc biệt phù hợp với môi trường có phần thưởng thưa thớt và yêu cầu nhiều bước thao tác liên tục.**
* **Đối với SAC + HER, hiệu năng vẫn đạt mức chấp nhận được nhưng kém ổn định hơn RHER. Khoảng cách đến mục tiêu dao động với biên độ lớn hơn, cho thấy HER truyền thống chưa xử lý tốt tính phụ thuộc theo chuỗi thời gian và khả năng lan truyền tín hiệu phần thưởng yếu hơn so với RHER.**
* **Trái lại, PPO mặc dù hội tụ sớm trong giai đoạn đầu nhưng đạt Goal\_Achieved thấp, đồng thời giá trị Distance không giảm đáng kể trong quá trình huấn luyện. Mô hình hầu như không tạo được hành vi gắp – nhấc vật thể rõ rệt, phù hợp với đặc điểm vốn có của PPO: hiệu quả mẫu thấp và khó thích ứng trong các nhiệm vụ điều khiển liên tục cần độ chính xác cao.**
* **DDPG và các biến thể của nó thể hiện sự thiếu ổn định rõ nhất. Các đường biểu diễn Distance xuất hiện nhiều dao động biên độ lớn, đặc biệt ở các epoch cuối. Riêng DDPG + RHER có Distance cao nhất, phản ánh việc thuật toán gặp khó khăn trong việc duy trì chính sách nhất quán và khai thác trải nghiệm trong không gian hành động liên tục nhạy cảm.**
* **Nhìn chung, SAC + RHER là phương pháp mang lại hiệu quả tốt nhất trong tác vụ gắp, với độ ổn định, khả năng tiếp cận mục tiêu chính xác và mức độ tận dụng trải nghiệm vượt trội. Ngược lại, PPO và DDPG thể hiện nhiều hạn chế, khó đạt được hành vi gắp vật ổn định trong môi trường phức tạp và phần thưởng thưa thớt.**

1. **KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

Nghiên cứu này đã khám phá những phương pháp tiếp cận sáng tạo để nâng cao khả năng điều khiển cánh tay robot thông minh và ước tính tư thế vật thể 6D. Bằng cách tích hợp Dense Fusion với Deep Fusion Transformer, chúng tôi đã đạt được độ chính xác cao hơn trong nhận dạng vật thể và ước tính tư thế trong môi trường phức tạp, đồng thời duy trì một mô hình gọn nhẹ và hiệu quả. Sự tích hợp này cho phép cánh tay robot nhận thức và hiểu môi trường xung quanh với độ chính xác cao hơn, một bước quan trọng hướng tới hoạt động mạnh mẽ và đáng tin cậy trong các tình huống thực tế.

Hơn nữa, việc áp dụng phương pháp Soft Actor-Critic (SAC) đã cải thiện đáng kể khả năng tương tác của cánh tay robot với các vật thể có hình dạng và kích thước khác nhau. Sự tiến bộ này cho phép cánh tay nắm và thao tác với nhiều loại vật thể hơn, tăng tính linh hoạt và khả năng thích ứng của nó. Bằng cách kết hợp RAC với DAC và kết hợp các kỹ thuật tiên tiến như RHER, PCGrad, MHER và CDP, chúng tôi đã tối ưu hóa hơn nữa quá trình học tập và ổn định việc thực hiện các nhiệm vụ tuần tự. Sự kết hợp các phương pháp này dẫn đến một hệ thống điều khiển mạnh mẽ và hiệu quả hơn, có khả năng xử lý các chuỗi hành động phức tạp với độ ổn định và độ chính xác được cải thiện.

Những đóng góp của nghiên cứu này đại diện cho một bước tiến đáng kể trong lĩnh vực điều khiển robot và ước tính tư thế 6D. Bằng cách cung cấp các giải pháp mạnh mẽ và hiệu quả cho các ứng dụng thực tế phức tạp, công trình này mở đường cho các hệ thống robot tinh vi và đáng tin cậy hơn. Các thuật toán và kỹ thuật được phát triển có tiềm năng được áp dụng trong các lĩnh vực khác nhau, bao gồm tự động hóa công nghiệp, hậu cần và thậm chí cả chăm sóc sức khỏe, nơi mà việc thao tác robot chính xác và thích ứng là rất quan trọng.

Trong công việc tương lai, chúng tôi hướng tới việc tăng cường sự đa dạng của các vật thể và tăng độ phức tạp của các nhiệm vụ để cải thiện hơn nữa các thuật toán. Điều này sẽ bao gồm việc cho cánh tay robot tiếp xúc với nhiều loại vật thể và tình huống hơn, thúc đẩy ranh giới khả năng của nó và dẫn đến khả năng điều khiển cánh tay robot thông minh và linh hoạt hơn nữa. Bằng cách liên tục thử thách và tinh chỉnh các thuật toán của mình, chúng tôi cố gắng tạo ra các hệ thống robot có thể tích hợp liền mạch vào môi trường của con người và thực hiện nhiều loại nhiệm vụ với tính tự chủ và hiệu quả cao hơn.

# CHƯƠNG 5: TÀI LIỆU THAM KHẢO

[1] Han, D., Mulyana, B., Stankovic, V., Cheng, S.: A survey on deep reinforcement learning algorithms for robotic manipulation. Sensors 23(7) (2023) <https://doi.> org/10.3390/s23073762

[2] Satheeshbabu, S., Uppalapati, N.K., Fu, T., Krishnan, G.: Continuous control of a soft continuum arm using deep reinforcement learning. In: 2020 3rd IEEE International Conference on Soft Robotics (RoboSoft), pp. 497–503 (2020). https://doi.org/10.1109/RoboSoft48309.2020.9116003

[3] Shahid, A.A., Piga, D., Braghin, F., Roveda, L.: Continuous control actions learning and adaptation for robotic manipulation through reinforcement learning. Autonomous Robots 46(3), 483–498 (2022) <https://doi.org/10.1007/>s10514-022-10034-z

[4] Marullo, G., Tanzi, L., Piazzolla, P., Vezzetti, E.: 6d object position estimation from 2d images: a literature review. Multimedia Tools and Applications 82(16), 24605–24643 (2023) https://doi.org/10.1007/s11042-022-14213-z

[5] Brachmann, E., Krull, A., Michel, F., Gumhold, S., Shotton, J., Rother, C.: Learning 6d object pose estimation using 3d object coordinates. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) Computer Vision – ECCV 2014, pp. 536–551. Springer, Cham (2014)

[6] Zhuang, C., Li, S., Ding, H.: Instance segmentation based 6d pose estimation of industrial objects using point clouds for robotic bin-picking. Robotics and Computer-Integrated Manufacturing 82, 102541 (2023) https://doi.org/10.1016/j.rcim.2023.102541

[7] Zhang, L., Zhou, X., Liu, J., Wang, C., Wu, X.: Instance-level 6d pose estimation based on multi-task parameter sharing for robotic grasping. Scientific Reports 14(1), 7801 (2024) https://doi.org/10.1038/s41598-024-58590-x

[8] Stevˇsi´c, S., Christen, S., Hilliges, O.: Learning to assemble: Estimating 6d poses for robotic object-object manipulation. IEEE Robotics and Automation Letters 5(2), 1159–1166 (2020) <https://doi.org/10.1109/LRA.2020.2967325>

[9] Liu, Z., Wang, H., Liu, F.: Anchor-based 6d object pose estimation. In: 2021 IEEE 7th International Conference on Virtual Reality (ICVR), pp. 33–40 (2021). <https://ieeexplore.ieee.org/document/9483838>Ali

[10] He, Z., Feng, W., Zhao, X., Lv, Y.: 6d pose estimation of objects: Recent tech- nologies and challenges. Applied Sciences 11(1) (2021) https://doi.org/10.3390/app11010228

[11] Marchand, E., Uchiyama, H., Spindler, F.: Pose estimation for augmented reality: A hands-on survey. IEEE Transactions on Visualization and Computer Graphics 22(12), 2633–2651 (2016) https://doi.org/10.1109/TVCG.2015.2513408

[12] Sinha, A., Mishra, R.K., Jaiswal, S.: Robust and smooth nonlinear control of an industrial robot for automated pick and place. In: 2015 International Conference on Computing Communication Control and Automation, pp. 1–4 (2015). https://doi.org/10.1109/ICCUBEA.2015.11

[13] Nahavandi, S., Alizadehsani, R., Nahavandi, D., Lim, C.P., Kelly, K., Bello, F.: Machine learning meets advanced robotic manipulation. Information Fusion 105, 102221 (2024) https://doi.org/10.1016/j.inffus.2023.102221

[14] Lillicrap, T.P., Hunt, J.J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., Wierstra, D.: Continuous control with deep reinforcement learning (2019)

[15] James, S., Johns, E.: 3d simulation for robot arm control with deep q-learning. CoRR abs/1609.03759 (2016) 1609.03759

[16] Gu, S., Holly, E., Lillicrap, T., Levine, S.: Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In: 2017 IEEE International Conference on Robotics and Automation (ICRA), pp. 3389–3396 (2017). https://doi.org/10.1109/ICRA.2017.7989385

[17] Hong, S., Wei, Y., Ma, M., Xie, J., Lu, Z., Zheng, X., Zhang, Q.: Research of robotic arm control system based on deep learning and 3d point cloud target detection algorithm. In: 2022 IEEE 5th International Conference on Automation, Electronics and Electrical Engineering (AUTEEE), pp. 217–221 (2022)

[18] Shahid, A.A., Piga, D., Braghin, F., Roveda, L.: Continuous control actions learning and adaptation for robotic manipulation through reinforcement learning. Autonomous Robots 46(3), 483–498 (2022) <https://doi.org/10.1007/>s10514-022-10034-z

[19] Zhu, Y., Li, M., Yao, W., Chen, C.: A review of 6d object pose estimation. In: 2022 IEEE 10th Joint International Information Technology and Artificial Intelligence Conference (ITAIC), vol. 10, pp. 1647–1655 (2022). <https://doi.org/>10.1109/ITAIC54216.2022.9836663

[20] C¸ alli, B., Walsman, A., Singh, A., Srinivasa, S.S., Abbeel, P., Dollar, A.M.: Benchmarking in manipulation research: The YCB object and model set and benchmarking protocols. CoRR abs/1502.03143 (2015) 1502.03143

[21] Rennie, C., Shome, R., Bekris, K.E., Souza, A.F.D.: A dataset for improved rgbd- based object detection and pose estimation for warehouse pick-and-place. CoRR abs/1509.01277 (2015) 1509.01277

[22] Hodan, T., Michel, F., Brachmann, E., Kehl, W., Buch, A.G., Kraft, D., Drost, B., Vidal, J., Ihrke, S., Zabulis, X., Sahin, C., Manhardt, F., Tombari, F., Kim, T., Matas, J., Rother, C.: BOP: benchmark for 6d object pose estimation. CoRR abs/1808.08319 (2018) 1808.08319

[23] Rothganger, F., Lazebnik, S., Schmid, C., Ponce, J.: 3d object modeling and recognition using local affine-invariant image descriptors and multi-view spatial constraints. International Journal of Computer Vision 66(3), 231–259 (2006) https://doi.org/10.1007/s11263-005-3674-1

[24] Xiang, Y., Schmidt, T., Narayanan, V., Fox, D.: Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. CoRR abs/1711.00199 (2017) 1711.00199

[25] Wen, B., Yang, W., Kautz, J., Birchfield, S.: FoundationPose: Unified 6D Pose Estimation and Tracking of Novel Objects (2024)

[26] Tekin, B., Sinha, S.N., Fua, P.: Real-time seamless single shot 6d object pose prediction. CoRR abs/1711.08848 (2017) 1711.08848

[27] Kehl, W., Manhardt, F., Tombari, F., Ilic, S., Navab, N.: SSD-6D: Making RGB- based 3D detection and 6D pose estimation great again (2017)

[28] Wang, C., Xu, D., Zhu, Y., Mart´ın-Mart´ın, R., Lu, C., Fei-Fei, L., Savarese, S.: Densefusion: 6d object pose estimation by iterative dense fusion. CoRR abs/1901.04780 (2019) 1901.04780

[29] Pereira, N., Alexandre, L.A.: Maskedfusion: Mask-based 6d object pose detection. CoRR abs/1911.07771 (2019) 1911.07771

[30] Pereira, N., Alexandre, L.A.: MPF6D: masked pyramid fusion 6d pose estimation. CoRR abs/2111.09378 (2021) 2111.09378

[31] He, Y., Huang, H., Fan, H., Chen, Q., Sun, J.: FFB6D: A full flow bidirectional fusion network for 6d pose estimation. CoRR abs/2103.02242 (2021) 2103.02242

[32] He, Y., Sun, W., Huang, H., Liu, J., Fan, H., Sun, J.: PVN3D: A deep point-wise 3d keypoints voting network for 6dof pose estimation. CoRR abs/1911.04231 (2019) 1911.04231

[33] Zhu, H., Sun, X., Li, Y., Ma, K., Zhou, S.K., Zheng, Y.: DFTR: Depth-supervised Fusion Transformer for Salient Object Detection (2022)

[34] Zhou, J., Chen, K., Xu, L., Dou, Q., Qin, J.: Deep Fusion Transformer Network with Weighted Vector-Wise Keypoints Voting for Robust 6D Object Pose Estimation (2023)

[35] Yang, Z., Merrick, K., Jin, L., Abbass, H.A.: Hierarchical deep reinforcement learning for continuous action control. IEEE Transactions on Neural Networks and Learning Systems 29(11), 5174–5184 (2018)

[36] Gao, X., Yan, L., Li, Z., Wang, G., Chen, I.-M.: Improved deep deterministic policy gradient for dynamic obstacle avoidance of mobile robot. IEEE Transactions on Systems, Man, and Cybernetics: Systems 53(6), 3675–3682 (2023) https://doi.org/10.1109/TSMC.2022.3230666

[37] Cheng, Y., Zhao, P., Wang, F., Block, D.J., Hovakimyan, N.: Improving the robustness of reinforcement learning policies with l1 adaptive control. CoRR abs/2112.01953 (2021) 2112.01953

[38] Shukla, P., Pegu, M., Nandi, G.C.: Development of behavior based robot manipulation using actor-critic architecture. In: 2021 8th International Conference on Signal Processing and Integrated Networks (SPIN), pp. 469–474 (2021). https://doi.org/10.1109/SPIN52536.2021.9566102

[39] Pohlmeyer, E.A., Mahmoudi, B., Geng, S., Prins, N., Sanchez, J.C.: Brain-machine interface control of a robot arm using actor-critic rainforcement learning. In: 2012 Annual International Conference of the IEEE Engineering in Medicine and Biology Society, pp. 4108–4111 (2012). <https://doi.org/10.1109/EMBC.2012.>6346870

[40] Kim, M., Han, D.-K., Park, J.-H., Kim, J.-S.: Motion planning of robot manipulators for a smoother path using a twin delayed deep deterministic policy gradient with hindsight experience replay. Applied Sciences 10(2) (2020) https://doi.org/10.3390/app10020575

[41] Li, S., Wang, G., Tang, Q., Wang, L.: Balancing value underestimation and overestimation with realistic actor-critic. CoRR abs/2110.09712 (2021) 2110.09712

[42] Dang Thi, P., Nguyen Truong, C., Dau Sy, H.: Rac-sac: An improved actor-critic algorithm for continuous multi-task manipulation on robot arm control. In: Proceedings of the 12th International Symposium on Information and Communication Technology. SOICT ’23, pp. 824–830. Association for Computing Machinery, New York, NY, USA (2023). https://doi.org/10.1145/3628797.3628939

[43] Nauman, M., Cygan, M.: On the Theory of Risk-Aware Agents: Bridging Actor-Critic and Economics (2024). <https://arxiv.org/abs/2310.19527>

[44] Yang, Y., Yang, R., Lyu, J., Yan, J., Luo, F., Luo, D., Li, X., Li, L.: Multi-step

hindsight experience replay with bias reduction for efficient multi-goal rein-

forcement learning. In: 2023 International Conference on Frontiers of Robotics

and Software Engineering (FRSE), pp. 144–156 (2023). https://doi.org/10.1109/

FRSE58934.2023.00028

[45] Luo, Y., Wang, Y., Dong, K., Zhang, Q., Cheng, E., Sun, Z., Song, B.: Relay

Hindsight Experience Replay: Self-Guided Continual Reinforcement Learning

for Sequential Object Manipulation Tasks with Sparse Rewards (2022). https:

//arxiv.org/abs/2208.00843

[46] Yu, T., Kumar, S., Gupta, A., Levine, S., Hausman, K., Finn, C.: Gradient surgery for multi-task learning. CoRR abs/2001.06782 (2020) 2001.06782

[47] Zhao, R., Tresp, V.: Curiosity-driven experience prioritization via density estimation. CoRR abs/1902.08039 (2019) 1902.08039

[48] Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Abbeel, P., Zaremba, W.: Hindsight Experience Replay (2018)

[49] Zhao, P., Lai, L.: Analysis of knn density estimation. IEEE Transactions on Information Theory 68(12), 7971–7995 (2022) <https://doi.org/10.1109/TIT.2022.>3195870

[50] Li, S., Tang, Q., Pang, Y., Ma, X., Wang, G.: Balancing Value Underestimation and Overestimation with Realistic Actor-Critic (2022)

[51] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal policy optimization algorithms. CoRR abs/1707.06347 (2017) 1707.06347

[52] Alex, Yarin, Roberto: Multi-Task Learning Using Uncertainty to Weigh Losses for Scene Geometry and Semantics (2017). <https://arxiv.org/abs/1705.07115>

[53] Baijong, Feiyang, Yu., Ivor W.: Reasonable Effectiveness of Random Weighting: A Litmus Test for Multi-Task Learning (2021). <https://arxiv.org/abs/2111.10603>

[54] Ozan Sener and Vladlen Koltun: Multi-task learning as multi-objective optimization (2018). <https://arxiv.org/abs/1810.04650>

[55] Bo Liu, Xingchao, Xiaojie, Peter Stone, Qiang Liu: Conflict-Averse Gradient Descent for Multi-task Learning (2021). <https://arxiv.org/abs/2110.14048>

[56] Bo Liu, Yihao, Peter Stone, Qiang Liu: FAMO: Fast Adaptive Multitask Optimization (2023). <https://arxiv.org/abs/2306.03792>

[57] Dmitry, Nikolay, Arseny, Anton: Independent Component Alignment for Multi-Task Learning (2023). <https://arxiv.org/abs/2305.19000>

[58] Meng Tian, Liagn, Marcelo., Gim: Robust 6D Object Pose Estimation by Learning RGB-D Features (2020). <https://arxiv.org/abs/2003.00188>

[59] Xiaohan, Xiangyu, Ningning, Jungong, Guiguang, Jian: RepVGG: Making VGG-style ConvNets Great Again (2021). <https://arxiv.org/abs/2101.03697>

[60] Zhen-Liang, Gui., Xiao-Hu, Zeng., Xiao-Liang, Chen, Yan, Rui., Zhen-Li: RAUNet: Residual Attention U-Net for Semantic Segmentation of Cataract Surgical Instruments (2019). <https://arxiv.org/abs/1909.10360>

[61] Ilya Loshchilov, Frank Hutter: Decoupled Weight Decay Regularization (2017). https://arxiv.org/abs/1711.05101

[62] Xingyu, Pan, Huan, Zhouchen, Shuicheng: Adan: Adaptive Nesterov Momentum Algorithm for Faster Optimizing Deep Models (2022). <https://arxiv.org/abs/2208.06677>

[63] Diederik P., Jimmy: Adam: A Metthod For Stochastic Optimization (2015). <https://arxiv.org/pdf/1412.6980>