Journal of KIIT. Vol. 24, No. 6, pp. 149-156, Jun. 30, 2026. pISSN 1598-8619, eISSN 2093-7571
http://dx.doi.org/10.14801/jkiit.2026.24.6.149

심층 강화 학습을 이용한 ROS 제어 시스템 구성 및 구현

구 본 근*

ROS Control System Configuration and Implementation using Deep Reinforcement Learning

Bongen Gu*

이 연구는 2024년 국립한국교통대학교 지원을 받아 수행하였음

요 약

신경망을 정책으로 사용하는 심층 강화 학습은 다양한 분야에서 결정문제를 해결하기 위해 사용된다. 신경망 기반의 정책을 이용하여 다음 행동을 결정하므로 심층 강화 학습이 로봇 제어 분야에 활용되고 있다. 특히, 로봇 제어용 플랫폼인 ROS에 심층 강화 학습을 적용하려는 연구들이 있다. 본 논문은 로봇 제어를 위해 심층 강화 학습 기반 ROS 제어 시스템 구성을 제안하고, 구현한다. 본 논문에서 제안하는 심층 강화 학습 기반 ROS 제어 시스템은 ROS와 연계되는 강화 학습 모델, 에이전트, 중재자, 임무 관리자, ROS 동료로 구성되어 있다. 본 논문에서 제안한 제어 시스템 구성이 모델의 개선, 제어 대상 로봇 변경, 환경의 변경 등에 대응할 수 있음을 보이기 위해 환경과 제어 방법이 다른 두 개의 응용을 구현한다.

Abstract

Deep Reinforcement Learning (DRL) which uses neural networks as a policy is used to solve decision-making problems in various fields. DRL is used for controlling a robot because it decides the next action using policy based on neural networks. Specially, many researchers focus on applying DRL to a robot application platform such as ROS. In this paper, we propose and implement a ROS control system based on DRL to control a robot. The ROS control system configuration proposed in this paper consists of reinforcement learning a model, an agent, an arbitrator, a mission manager and a ROS peer. To show that our proposed control system configuration can adapt to model updates, changes in target robots or changes in environment, we implemented two DRL-ROS applications with different in environments and control methods.

Keywords

reinforcement learning, DRL, ROS, robot control, reward, continuous action space

* 국립한국교통대학교 컴퓨터공학과 교수
- ORCID: https://orcid.org/0000-0002-7111-1440

· Received: May 06, 2026, Revised: May 22, 2026, Accepted: May 25, 2026

· Corresponding Author: Bongen Gu
Dept. of Computer Engineering
Korea National University of Transportation, Korea
Tel.: +82-43-841-5348, Email: bggoo@ut.ac.kr

Ⅰ. 서 론

현재 상태 또는 환경에 대한 관찰을 기반으로 에이전트의 행동을 결정하는 강화 학습은 마르코프 결정 과정(MDP, Markov Decision Process)을 기반으로 하고 있다[1][2]. 하지만 에이전트가 행동에 따른 다음 상태, 보상 등 환경에 대한 정보를 미리 알지 못하므로 행동에 따른 경험을 훈련한 신경망 기반의 정책을 통해 에이전트의 다음 행동을 결정하는 심층 강화 학습(DRL, Deep Reinforcement Learning)이 많은 연구와 응용에 사용되고 있다. 특히, 행동의 연속적인 결정 과정이라는 점에서 DRL은 자율주행, 로봇 제어 등의 분야에서 사용되고 있다. DRL을 로봇 제어에 적용한 연구 중 OpenAI Gymnasium에서 DDPG 알고리즘을 적용하여 신경망을 사전 훈련 시킨 후 연구자들이 제작한 소형 이동 로봇에 적용하여 성능을 평가한 연구가 있다[3].

ROS(Robot Operating System)는 로봇 관련 응용 및 개발을 위한 라이브러리와 소프트웨어 도구의 집합체이다[4]. 이 환경은 로봇 응용 시스템을 구성하는 다수의 노드 간 통신 방법 및 관련 기능을 제공한다. 노드 간 주요 통신 방법은 비동기 통신 기반의 토픽, 액션과 동기 통신 기반의 서비스 등이 있다. DRL의 정책 기반 행동 결정이 ROS 기반 로봇 제어를 통해 실제 환경 또는 가상 환경에서 실현하기 위해 DRL과 ROS를 결합한 연구가 있다[5]-[9].

ROS로 제어할 수 있는 RC 자동차에 DRL을 적용하여 사전 정보가 없는 환경에서 자율적으로 경로를 탐색하는 연구는 LiDAR 정보를 기반으로 Gazebo 시뮬레이션 환경에서 사전 훈련한 후 실제 환경에서 성능을 평가하였다[5]. Gazebo 시뮬레이션 환경에서 장애물을 회피하며 이동 로봇 경로 계획에 DRL을 적용한 연구는 RGB 이미지를 환경 정보로 사용하였다[6]. Turtlebot 등의 바퀴형 로봇이 충돌이 없이 주행할 수 있도록 PPO 강화 학습을 적용한 연구도 Gazebo 시뮬레이션 환경에서 사전 훈련한 후 실제 환경에 적용하였다[7]. 또, 자율 운항 선박 제어를 위해 장애물 회피 DQN 알고리즘을 적용한 연구는 시뮬레이션 환경에서 신경망을 훈련한 후 이를 기반으로 자율운항 장애물 회피 알고리즘 개발하고, 실제 소형 보트를 제작하여 장애물 회피 성능을 측정하여 DQN 기반 장애물 회피 알고리즘이 효과가 있음을 보였다[8]. 사족 보행 로봇(Quadruped robot) 제어에 DQN 기반 강화 학습과 ROS를 적용한 연구[9]는 사전 정보가 없는 환경에서 DRL, ROS, 컴퓨터 비전이 사족 보행 로봇 제어에 강건하고 확장이 가능한 해결 방법임을 보였다.

DRL과 ROS를 적용한 로봇 제어 관련 다양한 연구는 연구 목적에 따라 신경망 구성과 에이전트 알고리즘 등이 다양하지만, 정책 신경망, 에이전트, 환경과 상호 작용하는 인터페이스 등으로 구성된 DRL 부분과 실제 로봇 제어를 위한 ROS 연동 부분이 명확하게 구분되어 있지 않다. 따라서, 신경망, 에이전트 알고리즘, 제어 목표 또는 제어 대상 로봇 등에 변화가 있는 경우에 변화를 반영한 수정이 쉽지 않은 것으로 판단이 된다. 또, 신경망 구성 또는 알고리즘 등 한 연구의 결과물을 다른 연구에 적용하는 것도 쉽지 않다.

이러한 문제를 해결하기 위해 본 논문에서는 DRL의 구성 요소와 로봇 제어를 위한 ROS 연동 요소를 분리하고, 각 요소의 동작 종속 관계를 최소화함으로써 각 요소의 개별적 개선, 교체 등에 대응할 수 있는 시스템인 DRL 기반 ROS 시스템 구성을 제안한다. 또, 최소한의 변경만으로 목적과 환경이 다른 두 개의 제어 시스템을 시험 구현하여 본 논문에서 제안하는 시스템 구성이 환경, 제어 대상 등의 변화에 대응하여 제어 시스템을 구현할 수 있음을 보였다.

본 논문의 구성을 다음과 같다. 2장은 관련 연구로 ROS를 이용한 제어 분야에 DRL을 적용한 연구를 소개하고, 3장은 본 논문에서 제안하는 ROS 기반 제어 시스템의 구성을 기술한다. 4장에서는 본 논문에서 제안하는 DRL-ROS 제어 시스템을 구현한 내용을 기술한다. 마지막으로 5장에서는 본 논문에서 제안한 제어 시스템 구성의 의미와 결론을 기술하고, 향후 연구 관련 내용을 기술한다.

Ⅱ. 관련 연구

2.1 DRL과 ROS

강화 학습의 이론적 기반이 되는 MDP를 표현한 그림 1은 에이전트가 현재 상태 s_t에서 행동 a_t를 취한 결과로 다음 상태는 s_{t+1}이 되고, 이 시점에 획득하는 보상이 r_{t+1}임을 나타내고 있다. 즉, MDP는 현재의 상태와 에이전트가 취한 행동만으로 다음 상태가 결정됨을 의미한다[1][2].

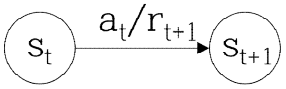


그림 1. 마르코프 결정 과정
Fig. 1. Markov decision process MDP

에이전트가 상태 st에서 행동 at를 결정하는 방법을 정책(Policy)이라고 한다. 에이전트는 환경에 대한 정보를 알지 못하는 상황에서 이전의 행동, 그 행동을 취한 결과인 다음 상태 및 보상 등에 대한 경험을 학습한 신경망을 이용하여 다음 행동을 결정한다.

로봇 응용 개발을 위한 통신 라이브러리와 도구로 구성된 ROS[4]에서 로봇 제어 모델은 제어 노드와 제어 대상 노드가 토픽을 통해 상호 작용하는 것이다. 즉, 제어 노드는 속도 및 방향 등 제어 관련 정보를 저장한 토픽을 게시하고, 위치, 방향, 주변 장애물 등 로봇의 상태 정보를 저장한 토픽을 구독한다. 반면, 제어 대상 노드는 제어 정보를 저장한 토픽을 구독하고, 각종 센서로부터 수집된 데이터를 기반으로 생성한 상태 정보를 저장한 토픽을 게시한다.

2.2 DRL을 적용한 ROS 응용

DRL을 이용하여 로봇의 다음 동작을 결정하는 ROS 기반 응용 연구 중에서 [5]는 LiDAR 데이터를 맵 이미지로 변환하여 환경 관측 데이터로 사용하고, 다섯 가지 이산 행동 공간에서 행동을 결정하였다. 이 연구는 사전 정보가 없는 환경에서 장애물을 회피하며 스스로 경로를 결정하는 방법을 제안하고, 실험하였고, 이를 위해 이산 행동 공간에서 활용할 수 있는 DQN 강화 학습 알고리즘을 사용하였으며, LiDAR 데이터를 이용한 보상 함수를 정의하였으며, 시뮬레이션 환경과 실 환경에서의 동작을 확인하였다. 이 연구에서 기술한 소프트웨어 구조로 Laser_Node, Algorithm_node, Pyserial_node를 제시하고 있어 확장성에 많은 제약이 있다고 판단된다.

[6]의 연구는 사전 정보가 없는 환경에서 장애물을 피하며, 목표 지점까지 이동하는 자율 주행 로봇을 위해 DQN의 변형인 D3QN과 무지개(Rainbow) 알고리즘을 구현하였으며, 시뮬레이션 환경에서 신경망을 훈련하고, 성능을 평가하였다. 이 연구는 깊이 카메라(Depth camera)로 캡처한 네 장의 연속 이미지를 환경 정보로 사용하였으며, 목표 지점을 향한 각도와 속도를 이용한 보상 함수를 사용하였다. 이 연구는 장애물을 피하며, 목표 지점까지 이동이라는 태스크 수행을 위한 알고리즘들을 비교, 분석하는 것에 중점을 두었으며, 보조적인 요소로 ROS를 사용하여, DRL 알고리즘과 ROS 요소 간의 구성에 대해 구체적으로 기술하지 않았다.

DRL을 ROS에 활용하는 많은 연구가 장애물을 피하며, 자율 주행하는 이동 로봇을 대상으로 하였지만, [8]의 연구는 LiDAR, GPS, IMU 등을 활용한 자율 운항 보트를 대상으로 하였다. 이 연구는 보트 제어에 ROS를 활용하였고, 장애물 회피를 위해 DQN 알고리즘을 사용하였으며, Gazebo를 사용하여 모델링한 가상 환경에서 신경망을 훈련한 후 소형 모형 보트에 적용하였다.

III. DRL 모델 기반 ROS 제어 시스템

기존의 DRL과 ROS를 이용한 많은 응용 구현은 다양한 요소에 대한 구성을 명확하게 기술하지 않아, 신경망 개선, 목표 태스크의 변경, 환경의 변화 등 다양한 변화요인에 대응하며 제어 시스템을 구현하는 것이 쉽지 않다는 문제가 있다.

본 논문에서 제안하는 DRL-ROS 시스템 구성은 DRL을 ROS에 적용할 때 응용 시스템의 각 구성 요소를 명확하게 분리함으로써 앞서 서술한 문제를 최소화한 ROS 기반 로봇 제어에 DRL을 적용할 수 있는 소프트웨어 구성이다. 그림 2는 본 논문에서 제안하는 DRL-ROS 시스템 구성을 나타낸 것으로 실선으로 표시한 다섯 개의 핵심 요소와 점선으로 표시한 한 개의 선택 요소를 포함하고 있다. 각 요소의 기능과 상호 작용은 다음과 같다.

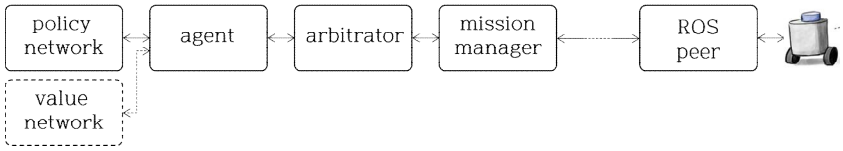


그림 2. DRL-ROS 시스템 구성
Fig. 2. Configuration of DRL-ROS system

정책 네트워크(Policy network)는 환경의 현재 상태에서 취할 행동을 결정하기 위한 정보를 제공하는 신경망이며, 기계 학습을 통해 최적화되어야 하는 훈련의 대상이다. 이 네트워크가 제공하는 정보는 에이전트가 취할 수 있는 각 행동의 확률 또는 행동을 위한 확률 분포 파라미터 등 응용에 따라 다를 수 있다. 알고리즘에 따라 선택적으로 사용될 가치 네트워크(Value network)는 상태의 가치를 추정하는 신경망으로서, 이 신경망도 기계 학습의 훈련 대상이 된다. 이 가치 네트워크는 에이전트가 수행하는 알고리즘에 따라 선택적으로 사용될 수 있다.

에이전트(Agent)는 다음 행동을 선택하고, 제공된 경험 정보를 기반으로 정책과 가치 네트워크의 파라미터를 조정하는 기능을 수행한다. 다음 행동을 선택하는 정책 $\pi(a_i|s_i,\theta)$ 및 파라미터를 조정하는 $\theta \leftarrow \theta + \alpha \cdot \nabla J(\theta)$은 에이전트가 수행하는 알고리즘에 따라 달라진다. 예를 들어, 이산 행동 공간 환경에서 정책 $\pi(a_i|s_i,\theta)$는 각 행동을 취할 확률이 될 수 있고, 연속 행동 공간(Continuous action space) 환경에서 정책은 어떤 확률 분포에서 행동을 샘플링하는 것이 될 수 있다.

중재자(Arbitrator)는 에이전트가 선택한 행동을 임무 관리자(Mission manager)에게 전달하고, 반환된 행동의 결과 즉, 다음 상태, 보상, 종료 여부 등에 대한 정보를 에이전트가 저장하도록 하며, 네트워크의 파라미터를 조정하는 기계 학습을 하도록 한다. 중재자는 사용된 알고리즘에 따라 에피소드 단위 또는 행동 단계 단위로 기계 학습을 진행할 수 있다.

임무 관리자는 ROS를 이용한 제어의 대상 로봇이 수행해야 하는 임무를 관리하고, DRL과 ROS 사이의 상호 작용을 위한 인터페이스 역할을 한다. DRL과 ROS 사이의 상호 작용을 위해 임무 관리자는 에이전트에서 결정한 행동을 ROS의 토픽으로 변환하여 게시한다. 이후 임무 관리자는 그가 게시한 토픽에 따라 동작을 수행한 로봇의 변경된 상태를 저장한 토픽을 구독하여, 그 토픽에 저장된 정보를 기반으로 계산한 보상을 중재자에게 전달한다. 보상 계산을 위한 보상 함수는 임무 관리자가 로봇의 임무 수행 과정을 제어하기 위한 수단으로 사용되며, 로봇의 임무에 따라 보상 함수가 달라져야 한다.

ROS 동료(ROS peer)는 임무 관리자가 게시한 토픽을 구독하여 로봇의 동작을 제어하고, 로봇의 동작 상태를 저장한 토픽을 게시하는 역할을 한다. 그림 3은 임무 관리자와 ROS 상대 사이에 진행되는 ROS 토픽 기반 상호 작용을 나타낸 것이다. DRL-ROS 시스템을 응용하는 분야마다 동작과 동작 후 환경 정보의 종류가 다르므로 토픽에 저장할 데이터를 위한 인터페이스를 별도 정의하여 사용할 수 있다.

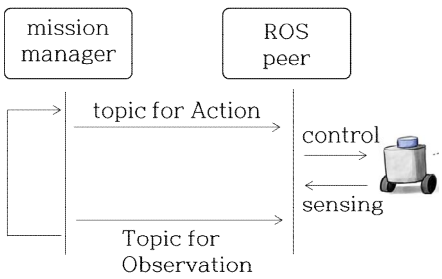


그림 3. 임무 관리자와 ROS 상대 사이의 상호작용
Fig. 3. Interaction between mission manager and ROS peer

IV. DRL-ROS 시스템 구성 구현

본 논문에서 제안한 DRL-ROS 제어 시스템 구성이 DRL 알고리즘의 변화, 환경의 변화 및 제어 대상의 변경 등에 대응할 수 있음을 보이기 위해 본 논문에서는 이동 로봇이 목적 위치로 이동하는 환경과 MountainCarContinuous-V0 환경에서 이동체를 제어하는 응용을 구현하였다.

4.1 구현 환경 및 공통 구현

서로 다른 두 ROS 응용이 DRL-ROS 제어 시스템의 구성 요소가 공유될 수 있음을 보이기 위해 본 논문에서는 그림 2의 요소 중 정책 네트워크, 가치 네트워크, 에이전트, 중재자를 공통 구현하였다.

표 1은 구현 환경을 나타내고 있다. 구현 환경의 시스템에 nVidia 사의 GPU를 위한 cuda 라이브러리가 설치되어 있으나, 본 구현에서 구현한 신경망의 훈련 대상 파라미터 수가 많지 않고, 구현의 목적이 성능 평가에 있지 않으므로 GPU 활용 여부가 실험에 영향을 주지 않았다. cuda 라이브러리가 설치되어 있지 않은 시스템 환경에서도 본 논문에서의 구현이 정상적으로 실행됨을 확인하였다.

본 논문에서 구현한 두 환경은 모두 연속 상태 공간을 갖는다. 본 논문에서는 별도로 구현한 인코더를 이용하여 상태를 특징 벡터로 인코딩한 후 이것을 정책과 가치 네트워크의 입력으로 사용하였다.

표 1. 구현 환경
Table 1. Implementation configuration

OS	Ubunut 24.06
Pytorch	2.11
cuda	12.2
numpy	1.26.4
ROS2	Jazzy

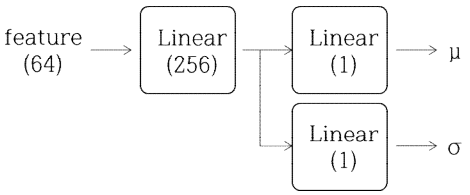


그림 4. 정책 네트워크의 레이어
Fig. 4. Policy network layers

인코더는 환경의 상태를 pytorch의 Linear(256), Linear(128), Linear(64) 세 단계의 완전 연결 신경망을 이용하여 크기가 64인 특징 벡터로 변환한다. 인코더 사용의 이점은 상태의 차원이 다르더라도 특징 벡터의 크기를 맞추면 정책 네트워크 등을 변경하지 않거나 최소화할 수 있다. 인코더의 출력인 특징 벡터를 입력으로 한 정책 네트워크는 그림 4에 보인 것과 같이 평균(μ)과 표준편차(σ)의 추정값을 출력하도록 구현하였다. 정책 네트워크가 추정한 두 값을 에이전트가 행동을 결정할 때 사용한다. 가치 네트워크는 특징 벡터를 입력으로 갖는 Linear(1) 계층을 이용하여 구성하였다.

A2C(Advantage Actor Critic) 알고리즘을 수행하는 에이전트는 정책 네트워크 출력인 평균(μ)과 표준편차(σ)를 파라미터로 한 Gaussian 확률 분포에서 샘플링하여 행동을 결정하고, 목적 함수의 미분값 $\nabla h(\theta)$[2]을 이용하여 정책 네트워크의 파라미터를 조정한다. 본 논문에서 구현한 중재자는 에이전트에서 결정한 행동을 임무 관리자에게 전달한 후 반환된 다음 상태 및 보상을 기반으로 에이전트가 정책 네트워크의 파라미터를 즉시 조정하도록 하였다.

그림 3의 임무 관리자와 ROS 동료는 응용마다 다르므로 각각 별도 구현하였고, 두 요소 간 상호작용을 위해 교환하는 토픽에 저장된 정보를 위한 자료형인 인터페이스도 별도 정의하였다.

4.2 목적 위치로 향하는 로봇 시뮬레이션 환경

그림 5는 DRL-ROS 시스템 구성을 시험 구현하기 위한 것으로 이동 로봇이 목적 지점까지 이동하도록 강화 학습을 활용하여 제어하는 시뮬레이션 환경을 나타낸 것이다. 강화 학습은 행동에 대한 보상을 통해 에이전트가 특정 목적 달성을 위한 최적 정책을 찾도록 한다. 이것은 보상 함수가 로봇의 임무 수행 목표를 반영할 수 있어야 함을 의미한다.

본 논문에서 구현한 임무 관리자는 목적 지점까지 이동이라는 임무를 로봇이 수행할 수 있도록 유도하기 위해 로봇과 목표 지점까지의 거리와 로봇이 향하고 있는 방향과 목표 지점의 각도를 이용하여 보상 함수를 정의하였다. 거리와 각도는 서로 다른 단위를 갖는 데이터이므로 표준화 과정을 통해 단위에 제거한 후 보상 함수를 적용하였다.

식 (1)은 본 논문에서 정의한 보상 함수이며, std_{dist}와 std_{rad}는 각각 로봇과 목표 지점까지의 거리 및 로봇의 방향과 목표 지점과의 상대적 각도를 표준화한 것이다. 타우(τ)는 두 정보의 중요도를 나타낸 것으로 거리보다는 방향이 중요하므로 본 논문에서는 이 값을 0.15로 설정하였다.

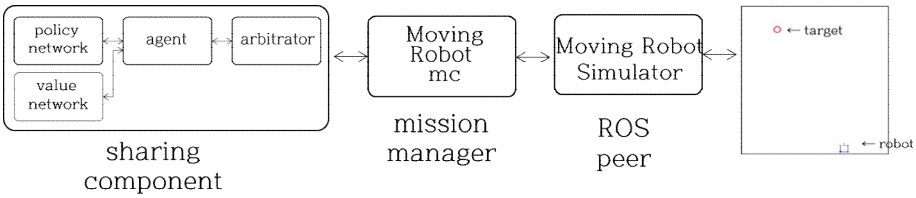
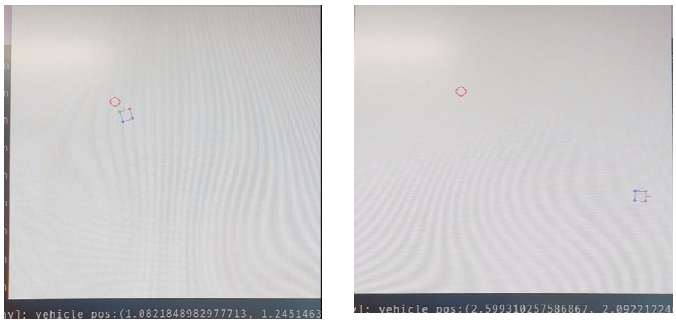


그림 5. 이동 로봇 시뮬레이션을 위한 DRL-ROS 구성
Fig. 5. DRL-ROS configuration for moving robot simulation



(a) (b)

그림 6. 이동 로봇 시뮬레이션을 위한 DRL-ROS 구현 (a) 성공 경우 (b) 실패 경우
Fig. 6. Implementation of DRL-ROS for moving robot simulation (a) Success case (b) Failure case

본 논문에서 구현한 임무 관리자는 중재자가 전달한 행동 정보를 저장한 토픽을 출판하고, 행동의 결과로 변경된 로봇의 위치, 방향 정보를 저장한 토픽을 구독한다.

$$R = \tau \times std_{dist} + (1-\tau) \times std_{rad} \quad (1)$$

그림 6은 DRL-ROS 제어 시스템 구성에 따라 구현한 이동 로봇 시뮬레이션의 동작한 화면을 캡처한 것이다. 그림 6(a)은 목적 지점 도착이라는 임무를 이동 로봇이 133회의 동작 스텝에 성공할 때 화면이며, 이때 획득한 전체 보상은 83.03이었다. 그림 6(b)은 이동 로봇이 1,000회의 동작 스텝 제한 내에 임무를 완료하지 못한 경우의 화면이다.

4.3 MountainCarContinuous-V0 환경

DRL-ROS 시스템 구성을 시험하기 위한 다른 응용 구현은 Gymnasium의 MountainCarContinuous 환경에서 ROS 기반으로 자동차를 제어한 것이다. 그림 7은 본 논문에서 구현한 DRL-ROS 구성을 나타낸 것이며, 환경에 종속적인 임무 관리자와 구독한 토픽을 이용하여 자동차를 제어하고, 새로운 상태와 보상을 저장한 토픽을 게시하는 ROS 동료 노드를 별도 구현하였다. 그림 7에서 공유 요소로 표시한 부분은 앞서 구현의 것을 수정하지 않고 그대로 이용하였다.

MountainCarContinuous 환경이 행동을 취한 결과인 다음 상태와 보상을 계산하므로 이 구현에서의 임무 관리자는 상대적으로 간단하게 구현할 수 있었다. 또, 두 요소가 교환하는 토픽도 응용에 맞게 정의하였다.

그림 8은 MountainCarContinuous-v0 환경에서 자동차를 제어하기 위해 구현한 DRL-ROS 제어 시스템이 동작한 화면을 캡처한 것이다. 그림 8(a)은 산 위의 깃발에 도착이라는 임무를 자동차가 841회의 동작 스텝에 성공한 것이며, 획득한 전체 보상은 16.68이었다. 그림 8(b)은 자동차가 1,000회의 동작 스텝 제한 내에 임무를 완료하지 못하고, 종료한 것을 나타내었다.

본 논문에서 제안한 DRL-ROS 시스템 구성은 다섯 개의 필수 요소와 한 개의 선택 요소로 구성되어 있다. 각 요소가 기능에 따라 명확하게 분리되어 있어, 본 논문에서 제안한 시스템 구성은 ROS를 이용한 로봇 제어에 DRL을 적용하는 응용에서 시스템의 구성 요소를 다양하게 조합할 수 있게 한다.

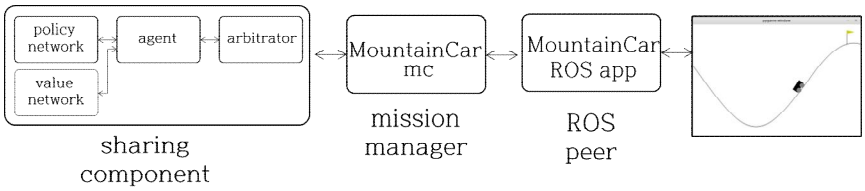
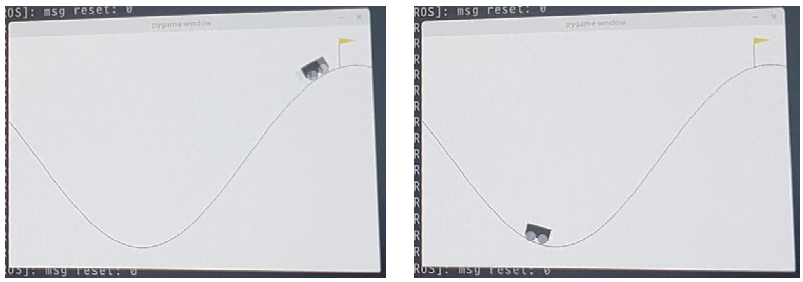


그림 7. MountainCarContinuous를 위한 DRL-ROS 구성
Fig. 7. DRL-ROS configuration for MountainCarContinuous



(a) (b)

그림 8. MountainCarContinuous-v0를 위한 DRL-ROS 구현 (a) 성공 경우 (b) 실패 경우
Fig. 8. Implementation of DRL-ROS for MountainCarContinuous-v0 (a) Success case (b) Failure case

예를 들어, 더 나은 성능을 위해 정책 네트워크의 세부 조정 또는 구성 변경 등의 신경망 개선이 다른 요소의 동작이나 구성에 영향을 받지 않고 독립적으로 이루어질 수 있다. 효과적인 신경망 훈련을 위해 에이전트 알고리즘 개선 또는 교체 등의 과정도 다른 구성 요소와 독립적으로 진행될 수 있다.

DRL-ROS 시스템을 구성하는 각 요소의 독립성은 서로 다른 응용이 이미 구현된 정책 네트워크, 에이전트, 중재자 등을 재사용하거나 공유할 수 있게 한다. 앞서 기술한 두 구현은 그 목적과 동작 방법이 다르지만, 정책과 가치 네트워크, 에이전트, 중재자는 수정이 없이 재사용되었다.

운반 로봇, 다축 관절 로봇 등 산업 현장의 로봇 제어를 위한 강화 학습 모델의 훈련은 비용, 안전성, 시간 등의 제약으로 실제 환경에서 진행되기 어렵다. 본 논문에서 제안한 DRL-ROS 시스템 구성은 제약이 있는 실 세계에서의 학습 모델 훈련 대신 시뮬레이션 환경에서 훈련을 진행한 후 ROS 동료 요소만 교체하면 실제 로봇 제어에 즉시 적용할 수 있다.

V. 결론 및 향후 과제

ROS는 로봇 응용 개발을 위한 라이브러리와 도구의 집합체로 소형 학습용 로봇부터 산업 현장의 로봇 제어에 활용되고 있다. DRL은 환경의 현재 상태에서 취할 최적 행동을 결정하는 정책을 신경망 훈련을 통해 찾는 것이다. DRL에서 결정한 행동을 ROS에서 로봇 제어에 활용하려는 연구들은 요소 간의 연동을 위한 시스템 구성을 구체적으로 기술하지 않았다.

본 논문에서는 ROS 기반의 로봇 제어에 DRL을 적용하기 위한 DRL-ROS 제어 시스템 구성을 제안하였다. 본 논문에서 제안한 시스템 구성은 정책 신경망, 에이전트, 중재자, 임무 관리자, ROS 동료 요소를 포함하고 있다. 기능에 따라 독립적으로 정의된 요소로 구성된 DRL-ROS 시스템은 구성 요소 간의 상호 작용을 통해 기능을 수행한다. 각 구성 요소가 상호 독립적이므로 필요에 따라 조합하거나 재사용할 수 있다는 장점이 있다.

DRL-ROS 시스템 구성이 실제 ROS 기반의 로봇 제어에 적용될 수 있음을 보이기 위해 본 논문에서는 정책 네트워크, 에이전트, 중재자를 재사용하여 구현한 두 개의 ROS 제어 응용을 구현하였다.

에이전트는 보상을 기반으로 정책 네트워크의 신경망을 훈련하고, 이를 기반으로 최종 임무 목적에 접근하는 행동을 결정하므로 임무 관리자의 보상

함수가 중요한 요소이다. 따라서, 향후 연구 과제는 자율 주행, 산업 로봇 등의 동작을 제어할 수 있는 임무 관리자의 보상 함수 결정 방법에 도출하는 것이다. 또, 소형 이동 로봇, 6축 관절 로봇 등 실제 환경의 로봇 제어에 본 논문에서 제안한 DRL-ROS 시스템을 적용하여 각 응용에 적절한 정책 네트워크, 에이전트 알고리즘을 개발하는 것이다.

References

[1] M. Ghasemi, A. Moosavi, and D. Ebrahimi, "A Comprehensive Survey of Reinforcement Learning: From Algorithms to Practical Challenges", arXiv preprint arXiv:2411.18892v2, pp. 2-4, Nov. 2024. https://doi.org/10.48550/arXiv.2411.18892.

[2] R. Atiensa, "Advanced Deep Learning with Keras", Packt Publishing, pp. 271-280, 2018.

[3] I. S. Zvonarev and Y. L. Karavaev, "Deep reinforcement learning for mobile robot control for motion toward a given position", Robotics and Autonomous System, Vol. 200, Art. no. 105392, Jun. 2026. https://doi.org/10.1016/j.robot.2026.105392.

[4] Robot Operating System, http://www.ros.org. [accessed: Apr. 15, 2026]

[5] S. Hossain, O. Doukhi, Y. Jo, and D. Lee, "Deep Reinforcement Learning-based ROS-Controlled RC Car for Autonomous Exploration in the Unknown Environment", 20th International Conference on Control, Automation and Systems(ICCAS2020), Busan, Korea, pp.1231-1236, Oct. 2020. https://doi.org/10.23919/ICCAS50221.2020.9268370.

[6] Q. Miguel, R. Jorge, and U. Victor, "Robot path planning using deep reinforcement learning", arXiv preprint arXiv:2302.09120, pp. 9-17, Feb. 2023. https://doi.org/10.48550/arXiv.2302.09120.

[7] H. Taheri, S. Hosseini, and M. Nekoui, "Deep Reinforcement Learning with Enhanced PPO for Safe Mobile Robot Navigation", arXiv preprint arXiv:2405.16266, pp. 2-5, May 2024. https://doi.org/10.48550/arXiv.2405.16266.

[8] J. H. Choi, "Development and verification of autonomous navigation obstacle avoidance DQN reinforcement learning algorithm based on robot operating system(ROS)", Master's thesis, Changwon National University, pp. 15-33, 2024.

[9] J. Soto, G. Lama, and J. Arcaya, "Quadruped Robot Prototype Based On Artificial Intelligence Using Deep Reinforcement Learning Through The ROS Framework", 2025 CHILECON, Valparaíso, Chile, Valparaíso, Chile, pp. 1-2, Oct. 2025. https://doi.org/10.1109/CHILECON66915.2025.11476093.

저자소개

구 본 근(Bongen Gu)

1991년 2월 : 인제대학교 전산학과(이학사)
1993년 2월 : 부산외국어대학교 대학원 컴퓨터공학과(공학석사)
1998년 2월 : 경북대학교 대학원 컴퓨터공학과(공학박사)
1998년 4월 ~ 현재 : 국립한국교통대학교 컴퓨터공학과 교수
관심분야 : 컴퓨터구조, 강화 학습, 로봇 제어, ROS