논문 21-46-10-15 The Journal of Korean Institute of Communications and Information Sciences '21-10 Vol.46 No.10
https://doi.org/10.7840/kics.2021.46.10.1667

심층강화학습기반 자율주행차량을 이용한 원형도로의 Stop-and-Go Wave 현상 해결 전략 연구

이 동 수◆, 권 민 혜°

Combating Stop-and-Go Wave Problem at a Ring Road Using Deep Reinforcement Learning Based Autonomous Vehicles

Dongsu Lee◆, Minhae Kwon°

요 약

인공지능 기술의 발전과 함께 자율주행기술의 발전 또한 가속화되고 있다. 본 연구에서는 심층 강화학습 알고리즘을 기반으로 한 자율주행차량을 이용하여 원형 도로에서 빈번하게 발생하는 Stop-and-go wave 현상을 해결하여 도로 흐름을 개선하고자 한다. 이를 위해 원형 도로에 적합한 마르코프 의사결정과정 모델을 제안한다. 자율주행차의 가속도 제어 정책 학습을 위해 Proximal Policy Optimization (PPO), Deep Deterministic Policy Gradient (DDPG), Twin Delayed DDPG (TD3) 알고리즘을 사용하고, 각 알고리즘별 학습 성능 및 주행 패턴의 차이를 확인한다. 시뮬레이션을 통해 자율주행차량을 도로에 배치함으로써 원형 도로의 Stop-and-go wave 현상을 해결할 수 있음을 확인하였다.

Key Words : Autonomous driving system, Deep reinforcement learning, Markov decision process, Proximal Policy Optimization (PPO), Deep Deterministic Policy Gradient (DDPG), Twin delayed DDPG (TD3), Stop-and-go wave, Traffic flow control

ABSTRACT

With the rapid development of artificial intelligence, autonomous driving has recently attracted considerable attention. This paper aims to use an autonomous vehicle to improve road flow by solving the stop-and-go-wave problem on a ring road. We design a special model of Markov decision process model to solve stop-and-go-wave and use three deep reinforcement learning algorithms to train autonomous vehicles: Proximal Policy Optimization (PPO), Deep Deterministic Policy Gradient (DDPG), and Twin Delayed DDPG (TD3). We then compare their driving patterns and performances. We confirmed that an autonomous vehicle on the ring road could control the flow of multiple non-autonomous vehicles with an extensive simulation study, thus successfully solving the stop-and-go wave problem.

※ 이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원(IITP-2021-0-00739, 분산/협력 AI 기반 5G+ 네트워크 데이터 분석 기능 및 제어 기술 개발)과 한국 연구재단(NRF-2020R1F1A1069182, 모바일 네트워크에서의 강화학습 기반 분산적 의사결정방법 연구)의 지원을 받아 수행된 연구임.

◆ First Author : Soongsil University School of Bio-medical System, movementwater@soongsil.ac.kr, 학생회원
° Corresponding Author : Soongsil University School of Electronic Engineering, minhae@ssu.ac.kr, 정회원
논문번호 : 202105-109-C-RN, Received May 24, 2021; Revised July 9, 2021; Accepted July 26, 2021

Ⅰ. 서 론

인공지능 기술의 도약과 함께 자율주행기술 분야는 대중과 연구자들 모두에게 흥미로운 주제 중 하나로 각광받고 있다. 미국자동차공학회(Society of Automotive Engineers)는 자율주행기술을 0~5 단계로 구분한 가이드를 제공한다. 0단계인 완전 비 자율주행부터 운전자 보조, 부분 자율주행, 조건부 자율주행, 고도 자율주행, 그리고 5단계인 완전 자율주행 기술까지 총 6단계로 구분할 수 있다[1]. 현재 국내외 기업에서 3단계 자율주행차량의 개발을 성공적으로 이루었으며 Waymo, GM, 현대기아 등은 4단계 자율주행차량의 개발을 진행 중에 있다[2]. 또한 한국은 세계 최초로 3단계 자율주행차량 안전기준을 발표하였으며 2020년 7월부터 3단계 차량의 출시 및 판매가 가능해졌다[3]. 한국과학기술평가원이 2019년에 보고한 바에 따르면 2020년부터 2035년까지 레벨 4단계의 자율주행차량 시장은 연평균 84.2%의 성장을, 레벨 3단계의 경우 연평균 33.6%의 성장이 전망되었다[4].

자율주행기술의 구성 요소에는 환경 및 위치 인식과 같은 인지 기술, 판단 기술, 그리고 제어 기술과 함께 탑승자에게 정보를 제공하는 인터페이스(interface)가 있다[5]. 3가지 요소들이 완벽하게 학습이 될 때 완전 자율주행차량이 상용화될 수 있으며, 이러한 연구는 인공지능의 발전에서 핵심적인 역할을 한 딥러닝(deep learning)을 통해 집중적으로 연구되고 있다.

자율주행기술의 연구는 역할에 따른 기능 모듈을 독립적으로 학습시키거나 구현하는 파이프라인 방식[6]과 전체의 과정을 한 번에 학습하는 엔드-투-엔드(end-to-end) 방식이 있다[7,8]. 먼저, 기능적 모듈화를 통한 파이프라인 방식은 주로 인지(perception), 계획(planning), 행동(action)의 모듈로 구성되며, 각 모듈은 독립적으로 학습된다[8]. 반면, 엔드-투-엔드 방식은 자율주행기술 요소의 기능적인 모듈화 과정 없이 자율주행의 과정 전체를 학습하는 방식이다. 가장 대표적인 학습 방법으로는 강화학습(reinforcement learning)이 있다.

강화학습은 학습 개체(agent)가 환경(environment)과 상호작용하며 시행착오를 통해 최적의 행동을 찾아내는 기계학습 방법의 하나이다. 강화학습에 딥러닝을 접목시킨 심층 강화학습(deep reinforcement learning)은 환경의 복잡성과 불확실성이 큰 경우에도 심층신경망을 이용하여 최적 정책 학습이 가능하다. 따라서, 자율주행기술과 같이 복잡한 현실 세계의 공학 문제에 널리 활용된다.

심층 강화학습을 위한 알고리즘은 구글 딥마인드(Google DeepMind)에서 제안한 Deep Q-Network (DQN)[9]을 시작으로 계속해서 발전을 이루고 있다. 그러나 자율주행기술에는 DQN 을 제외한 최신 알고리즘이 적용된 연구 사례 및 알고리즘 간 성능 비교가 드물게 이루어지고 있다. 특히, 알고리즘에 따라 행동전략이 다르게 학습될 수 있기에, 주행 패턴이 중요한 자율주행차량의 경우 알고리즘 간 비교 연구가 필수적이다.

본 연구에서는 자율주행차량을 이용한 도로 흐름 제어성능 평가를 위해 원형 도로에 다수의 차량을 배치한 환경을 사용한다. 이와 같은 환경에서는 특정 차량이 감속하게 되면 반응 시간 지체 현상(reaction time delay)에 의해 뒤따라오는 차들이 연달아 감속하는 현상이 발생한다. 이는 실제 도로에서 특별한 원인 없이 교통이 정체되는 유령 체증의 원인이 될 수 있다. 또한 원형 도로 환경의 특성상 감속 현상이 발생하는 도로의 반대편에는 가속하는 구간이 생긴다. 결과적으로 가속과 감속에 의해 전체 차량의 속도가 물결치는 반복적인 현상이 관찰된다. 즉, 원형도로에서 특정 차량의 감속으로 인해 전체 차량의 가속과 감속이 반복적으로 나타나는 현상을 Stop-and-go wave[10,11] 라고 한다. 이를 최소화할 수 있도록 학습된 자율주행차량을 도로에 배치하여, 전반적인 도로 흐름을 원활하게 하는 것이 본 논문의 목적이다.

본 논문에서는 자율주행 차량 학습을 위해 심층 강화학습 알고리즘인 PPO[12], DDPG[13], TD3[14]를 이용한다. 각 알고리즘을 통해 학습시킨 자율주행차량의 주행패턴 및 성능을 비교 분석하여 가장 효율적인 주행을 지향하는 알고리즘을 확인하고자 한다.

본 논문의 구성은 다음과 같다. II장에서 본 연구와 관련된 선행연구를 소개한다. III장은 본 연구에서 해결하고자 하는 문제에 대한 마르코프 의사결정 과정 모델링 및 알고리즘을 제안한다. IV장에서는 심층 강화학습 알고리즘에 대한 구성을 소개한다. V장에서는 시뮬레이션 설정 소개 후 성능을 평가한다. 마지막으로 VI장에서는 결론을 다룬다.

Ⅱ. 선행 연구

2.1 심층 강화학습 알고리즘

구글 딥마인드의 알파고(Alpha-Go)[15]의 대표 기술인 심층 강화학습은 로봇공학(robotics)[16], 자율주행

기술[17,18], 화학반응의 최적화[19] 등 다양한 분야에 적용이 되고 있다. 특히, 환경의 변화를 정확히 예측하기 어려운 경우에도 활용 가능한 Model-free 강화학습[20]의 경우에는 자율주행차량 제어를 포함한 공학 시스템에 활발히 활용되고 있다.

Model-free 강화학습 알고리즘은 크게 두 가지 방식으로 분류할 수 있다. 가치 기반(value-based)[9]의 알고리즘과 정책 기반(policy-based)[21] 학습 알고리즘이다. 먼저 Q-learning으로 대표되는 가치 기반의 알고리즘은 상태 정보와 행동의 조합에 대한 가치를 평가해주는 Q-function을 학습하는 것을 목적으로 한다. DQN의 경우는 이 과정에서 딥 러닝 기반의 함수 근사화 기술이 활용된다. 학습 개체는 주어진 환경 상태에서 Q-value를 최대화할 수 있는 행동을 선택하여 수행한다. 반면, 정책 기반 학습은 별도의 가치평가 과정 없이 상태정보를 입력으로 받아 행동을 출력해주는 정책을 학습한다. 대표적으로 REINFORCE[22]가 있다. 추가적으로, 가치 기반 학습과 정책 기반 학습 방식을 결합한 액터-크리틱(actor-critic) 구조도 개발되었다. 액터-크리틱 방법은 정책을 근사하는 액터 네트워크와 함께 가치함수를 근사하는 크리틱 네트워크를 사용하는 방법이다. 본 논문에서 다룰 PPO, DDPG와 TD3가 대표적이다.

2.2 심층 강화학습 기반의 자율주행 기술 연구

자율주행 기술의 연구는 활발히 진행되고 있다. 자율주행차량의 인지, 계획 및 의사 결정 분야의 발전을 통해 이미 단순한 구조의 도로를 주행하는 등의 단순 주행기능은 크게 향상되었다[7]. 고속도로 환경에서 교통 시나리오별 지능형 주행 기술은 1993년 Varaiya에 의해 세분화 되었다[23]. 하지만 사전에 정의된 환경에 국한된 알고리즘을 사용하는 방법은 일반화된 도로 환경과 주행 시나리오에 적용될 수 없다는 한계점이 있다. 특히, 실제 도로에서는 다른 차량들과의 관계에서 오는 환경의 복잡성 및 불확실성으로 인해 기존에 정의한 상황에서 최적의 의사결정을 하지 못한다는 한계점이 존재한다.

이러한 한계를 극복하기 위해 심층 강화학습을 적용한 Model-free 기반의 엔드-투-엔드 방식이 새로운 패러다임으로 떠오르며 활발히 연구되고 있다[6]. 대표적으로, DQN 알고리즘의 심층 신경망에 Long Short-Term Memory (LSTM)을 결합한 DRQN이 제안되어 Vdrift 프레임워크상의 환경에서 자율주행차량을 학습시키는 연구가 진행되었다[24]. 최근에는 DQN 기반의 연구 뿐 아니라 최신 알고리즘을 적용한 연구 또한 점진적으로 진행되고 있다. PPO를 실제 차량의 자율주행에 적용하기도 하고, TORCS 프레임워크를 이용하여 DDPG를 적용한 연구도 존재한다[17,25].

앞서 언급한 선행 연구를 통해 DQN부터 최신 알고리즘까지 자율주행 기술에 활발히 적용됨을 확인할 수 있다. 하지만 각 알고리즘별 성능 비교에 대한 연구는 상대적으로 매우 적다. 그리고 대부분의 연구는 특정한 도로 환경에서의 정상 주행을 목표로 하고 있고, 본 논문에서 다루는 자율주행차를 활용한 정체 현상 해결에 관한 연구는 미비했다. 따라서 본 연구에서는 차량 밀집 환경에서 도로의 흐름을 제어할 수 있는 MDP 모델을 제안하고자 한다. 또한, 심층 강화학습 알고리즘인 PPO, DDPG, TD3의 성능을 비교하여 원형 도로에서 가장 효율적인 주행패턴을 학습하는 알고리즘을 확인하고자 한다.

2.3 Stop-and go wave 현상 해결 자율주행 기술 연구

본 연구를 통해 해결하고자 하는 문제인 Stop-and-go wave 현상에 대한 많은 선행 연구 역시 존재한다. 고전적인 제어 이론기반의 알고리즘을 실제 차량에 적용한 연구[26]는 한 대의 차량만으로 전체 차량흐름 제어가 가능하다는 것을 확인하였다. 다양한 제어 이론 기반 알고리즘을 시뮬레이션을 통해 비교 분석한 연구[27]에서는 Intelligence Driving Model (IDM), Adaptive Cruise Control (ACC), Cooperative ACC (CACC)의 각 알고리즘 별 성능특성을 분석하였다. 고전적인 제어이론 기반 알고리즘은 정확한 역학 모델에 따라 사전에 정의된 시나리오에서는 최적의 성능을 이끌어 낼 수 있다는 장점이 있다. 하지만, 예측하기 어려운 많은 시나리오를 모두 고려하거나, 역학 모델을 정확하게 사용하는 것은 매우 어렵다. 이러한 한계를 극복하기 위해 본 연구와 같이 심층 강화학습 방법을 적용한 연구가 진행되었다[28,29]. 심층 강화학습을 사용하는 경우에는 환경상태에 대한 정보가 필요한데, 도로에 존재하는 모든 차량의 정보를 상태정보로써 자율주행차량이 이용할 수 있다는 가정이 존재하는 경우가 있다[28]. 그러나 실제 주행 환경에서 개체는 주변차량 정보만 인지할 수 있기 때문에 이러한 가정은 큰 한계로 다가올 수 있다. 따라서 본 연구에서는 자율주행차량이 선두차량만 관측 가능하도록 한정하여 선행 연구들의 한계점을 해결하였다.

Ⅲ. Stop-and-go wave 현상 해결을 위한 심층 강화학습 기반 자율주행차량 학습 전략

3.1 도로 환경

본 연구에서는 다수의 차량이 존재하여 정체 현상이 빈번하게 나타나는 원형 도로를 다룬다(그림1). 원형 도로에서는 한 차량이 감속하게 되면 뒤따른 차들이 연쇄적으로 감속을 하게 되는 한편, 도로의 반대편에서는 가속하는 구간이 생기기 때문에 Stop-and-go wave 현상이 빈번하게 발생한다.

원형 도로 내 전체 차량의 집합 E는 다음과 같이 정의한다.

$$E=\{e_1,\cdots,e_{j-1},e_j,e_{j+1},\cdots,e_N\}$$

여기서 j번째 차량인 e_j는 자율주행차량을 의미한다. 자율주행차량을 제외한 모든 차량은 비 자율주행차량으로 $e_{i,i\neq j}$로 정의한다. 차들은 도로에서 반시계방향으로 운동하며 차량의 인덱스는 시계 방향으로 정의된다. 즉, $j-1$번째 차량 e_{j-1}은 e_j의 선두 차량을 의미한다. 본 연구에서 자율주행차량의 수는 1대로 한정하며, 전체 차량의 수는 $|E|=N$이다. 따라서, 비 자율주행차량의 수는 $N-1$대 이다. 원형 도로의 길이는 l로 정의한다. 차량의 위치는 도로 내 기준점으로부터 이동한 실제 거리로 나타낸다. 시간 t에서 i번째 차량 e_i의 위치는 $d_{t,i}$로 정의한다. 위치 가능한 공간의 집합은 $D=\{d_{t,i}\in\mathbb{R}\mid 0\leq d_{t,i}\leq l\}$로 정의한다.

그림 1. 원형도로에서의 멈춤-가속 반복 현상
Fig. 1. Stop-and-Go wave on ring network

3.2 마르코프 결정 과정(Markov Decision Process) 모델 제안

자율주행 차량 e_j는 확률 모델인 마르코프 의사결정 과정(Markov Decision Process; MDP)을 통해 가속도를 제어한다. MDP 확률 모델은 $<S,A,R,\gamma>$의 튜플(tuple) 형태로 표현할 수 있다. S는 개체가 위치할 수 있는 유한한 상태 공간(state space)을 의미하며 A는 개체가 취할 수 있는 모든 행동이 포함된 유한한 행동 공간(action space)을 의미한다. R은 환경으로부터 개체가 받게 되는 보상공간이다. 마지막으로 γ는 감가율(discount factor)으로 즉각적인 보상과 미래에 받게 되는 지연 보상 사이의 중요도를 결정하는 변수이며 $0\leq\gamma\leq 1$을 만족한다.

시간 t의 자율주행 차량 e_j의 상태정보 $s_{t,j}\in S$는 수식 (1)와 같이 정의한다.

$$s_{t,j}=\left[v_{t,j},v_{t,j}-v_{t,j-1},f(d_{t,j-1}-d_{t,j})\right]^T \tag{1}$$

여기서 $v_{t,j}$는 시간 t의 자율주행차량 e_j의 속도를 의미하며, $v_{t,j}-v_{t,j-1}$는 시간 t의 자율주행 차량 e_j와 선두차량 e_{j-1}의 상대 속도를 의미한다. $f(d_{t,j-1}-d_{t,j})$는 시간 t의 자율주행 차량 e_j와 선두차량 e_{j-1}의 상대 위치를 의미하며 길이 l의 원형 도로라는 특성을 반영하여 함수 $f(x)$는 다음과 같이 정의된다.

$$f(x)=\begin{cases}x & x\geq 0\\ l+x & x<0\end{cases} \tag{2}$$

본 연구에서는 현실적인 문제 설정을 위해 자율주행차량 e_j의 상태 정보 $s_{t,j}$에 도로 내 모든 차량이 아닌 선두 차량 e_{j-1} 한 대만 포함하였다. 따라서 상태 정보의 차원은 $s_{t,j}\in\mathbb{R}^{3\times 1}$로 정의된다.

그림 2를 통해 자율주행차량 e_j가 관측 가능한 정보를 확인할 수 있다. 이때 빨간색 차량은 자율주행차량 이며, 파란색 차량은 자율주행차량의 선두차량 e_{j-1}으로 관측 가능한 비 자율주행차량을 나타낸다. 흰색 차량은 관측 불가능한 비 자율주행차량을 나타낸다.

자율주행차량 e_j가 취할 수 있는 행동은 가속도 $a_{t,j}\in A$로 정의된다. 유한한 행동 공간 A는 물리적으로 한정하는 최소가속도 $a_{\min}$부터 최대가속도 $a_{\max}$까지의 실수 집합

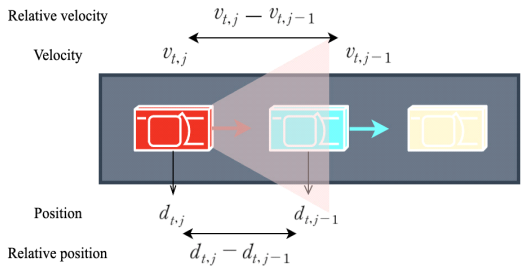


그림 2. 자율주행차량의 관측 가능 상태 정보
Fig. 2. The state information of the autonomous vehicle

$A = \{a_{t,j} | a_{\min} \leq a_{t,j} \leq a_{\max}\}$로 정의한다. 여기서 $a_{\min}$은 음의 실수이며 $a_{\max}$는 양의 실수이다. $a_{t,j} < 0$인 경우는 브레이크가 작동되는 것을 의미하며, $a_{t,j} > 0$인 경우에는 액셀이 작동되는 것을 의미한다.

밀집된 차량 환경에서 Stop-and-go wave 현상 방지를 위해 설정한 보상 함수 $r_{t,j} | N R$은 다음과 같이 정의한다.

$$r_{t,j} = \frac{\eta_1}{N}\sum_{i=1}^{N} v_{t,i} + \eta_2(\xi - |a_{t,j}|) \quad (3)$$

보상함수는 두 항으로 구성된다. 첫 번째 항 $\frac{\eta_1}{N}\sum_{i=1}^{N} v_{t,i}$ 은 보상 항(reward term)으로 도로 내 전체 차량의 속도의 평균으로 설정하였다. 이를 통해 도로의 평균 속도를 증가시키는 방향으로 학습하여 정체 현상을 방지할 수 있도록 한다. 두 번째 항 $\eta_2(\xi - |a_{t,j}|)$은 처벌 항(penalty term)으로 임계값(threshold) ξ와 자율주행차량 e_j의 가속도 $a_{t,j}$의 절댓값 사이의 차이를 이용한다. 만약, $\xi = 0$ 이면, 자율주행차량이 가속하는 행위에 대한 처벌이 될 수 있다. 처벌 항은 Stop-and-go wave 현상의 원인인 특정 차량의 급가속 및 급감속을 막아 등속 주행 할 수 있도록 해준다. 마지막으로 η_1, η_2는 각 항의 스케일의 조절을 위해 사용되는 파라미터이다.

이렇게 설계한 보상함수는 자율주행차량이 특정 속도에 도달한 뒤 등속 주행을 할 수 있도록 한다. 즉, 정체 현상 및 연쇄적인 지연 현상을 방지하고 결과적으로 멈춤과 가속의 반복 현상의 감소를 기대할 수 있다.

자율주행차량의 최종 목표는 미래의 누적 보상을 최대로 만드는 최적의 정책 π^*을 학습하는 것이다. 이때 정책 π은 상태 $s_{t,j}$에서 가능한 행동 $a_{t,j}$의 분포를 나타내며, 정책은 확률적 $a_{t,j} \sim \pi(s_{t,j})$ (stochastic policy)일 수도 있고 결정론적 $a_{t,j} = \mu(s_{t,j})$ (deterministic policy)[1]일 수도 있다. 정책을 결정하기 위해서는 상태나 상태-행동 조합의 가치를 평가해야 하기에 상태 가치 함수 $V_{t,j}(s)$ (state-value function) 혹은 행동 가치 함수 $Q_{t,j}(s,a)$ (action-value function or Q-function)을 사용한다. 가치함수는 다음과 같이 정의 한다.

$$V_{t,j}(s) = E[\sum_{k=0}^{T-t}\gamma^k r_{t+k,j} | s_{t,j} = s]$$

$$Q_{t,j}(s,a) = E[\sum_{k=0}^{T-t}\gamma^k r_{t+k,j} | s_{t,j} = s, a_{t,j} = a]$$

3.3 강화학습 기반 자율주행차량 학습 알고리즘 제안

본 논문에서 제안한 MDP 문제를 해결하기 위한 학습 과정은 알고리즘 1을 통해 확인할 수 있다. 우선 자율주행차량을 학습시키기 위해 심층 강화학습 알고리즘을 선택한다. 그리고 시뮬레이터(simulator)와 알고리즘 각각의 파라미터를 설정한다. 학습이 시작되기 전 알고리즘의 심층 신경망을 구성하는 파라미터 Θ를 초기화한다. 파라미터 설정을 할 때 알고리즘이 만약 DQN 기반의 방식이라면 목표 네트워크(target network)를 위한 파라미터 Θ'도 초기화한다. 전체 K번의 에피소드를 진행할 때 매 에피소드마다 환경 및 알고리즘이 새롭게 초기화되며 각 에피소드는 T Time steps으로 구성된다. 매 Time step t마다 개체는 상태 정보 $s_{t,j}$를 얻은 뒤 액터 네트워크에서 근사된 정책 $\pi_{\Theta_{actor}}$을 통해 행동 $a_{t,j}$을 선택하여 수행한다. $a_{t,j}$ 을 수행하면 개체는 보상 $r_{t,j}$ 와 함께 다음 상태 정보 $s_{t+1,j}$ 를 얻는다. 이렇게 수집한 일련의 정보 $< s_{t,j}, a_{t,j}, r_{t,j}, s_{t+1,j} >$을 이용해 네트워크를 업데이트하는데 사용한다. 만약 DDPG 혹은 TD3와 같이 Off-policy 방법을 사용한다면 경로정보를 Replay buffer B에 저장하여 네트워크의 업데이트에 사용한다[9]. 네트워크의 업데이트 및 평가 방식에 사용되는 목적함수(objective function)는 각 알고리즘마다 다르며 각각의 목적함

1) 확률적 정책과 구분을 위해 결정론적 정책의 경우 π가 아닌 μ로 표기한다.

수를 통해 네트워크 및 목표 네트워크를 업데이트 한다. 본 논문에서 사용한 각 알고리즘의 목적함수는 다음 절에서 다루고자 한다.

Ⅳ. 심층 강화학습 알고리즘의 네트워크 구조 및 목적함수

4.1 Proximal Policy Optimization (PPO)

PPO의 네트워크 집합 $\Theta=[\theta^{\pi},\theta^{VF}]$은 액터 네트워크$\theta^{\pi}$와 크리틱 네트워크 θ^{VF}로 구분할 수 있다. PPO 의 액터 네트워크를 업데이트하기 위한 목적함수는 클립된 대리 목적함수(clipped surrogate obj ective function) 라고 불리며 다음과 같은 식을 갖는다.

$$L_t^{CLIP}(\theta^{\pi})=E_t\left[\min(r_t(\theta^{\pi})\hat{A}_t, clip(r_t(\theta^{\pi}),1-\epsilon,1+\epsilon)\hat{A}_t)\right] \tag{4}$$

θ^{π}는 정책을 근사하는 액터 네트워크의 파라미터이다. $r_t(\theta)=\frac{\pi_{\theta}(a_{t,j}|s_{t,j})}{\pi_{\theta_{old}}(a_{t,j}|s_{t,j})}$는 이전 정책 $\pi_{\theta_{old}}$과 현재 정책 π_{θ} 간의 확률비를 나타낸 값이며, $\hat{A}_t$ (5)는 추정된 이득 함수(advantage function)이다. 마지막으로 ϵ은 클리핑의 기준을 정하는 파라미터이다.

PPO에서 크리틱 네트워크를 업데이트하기 위해 사용되는 목적함수는 다음과 같다.

$$L_t(\theta^{VF})=\max(L_t^{VF_1}, L_t^{VF_2}) \tag{6}$$

크리틱 네트워크의 목적함수 L_t^{VF}는 두 개의 목적함수 $L_t^{VF_1}$ 와 $L_t^{VF_2}$ 중 더 큰 값을 이용하여 나타낼 수 있다. $L_t^{VF_1}=(V_{\theta^{VF}}(s_{t,j})-V_t^{targ})^2$는 목표 가치함수 V_t^{targ}와 크리틱 네트워크의 파라미터 θ^{VF}로 근사되는 현재의 가치함수 $V_{\theta^{VF}}(s_{t,j})$의 차, 즉 오차의 제곱으로 나타낸다. $L_t^{VF_2}=(V_t^{CLIP}-V_t^{targ})^2$는 V_t^{CLIP} 수식 (7)과 목표 가치함수 V_t^{targ}의 오차의 제곱을 통해 나타낸다. 이때, V_t^{pred} 는 예측된 가치함수 즉, 업데이트 이전의 네트워크에서 얻은 가치함수의 출력값을 의미한다.

이와 같은 목적함수의 근사는 PPO 알고리즘의

Algorithm 1. Deep reinforcement learning based acceleration control algorithm for autonomous vehicles

Input: Choose a type of algorithm to train the agent among PPO, DDPG and TD3
Output: network parameters Θ = $[\Theta_{critic},\Theta_{actor}]$
Require: total episode K, time step of an episode T_E, soft update rate τ , range of road length $[l_{\min},l_{\max}]$, the number of vehicles on road N
if algorithm is TD3 **then**
  Set policy delay parameter ϕ
end if
Initialize network parameters Θ = $[\Theta_{critic},\Theta_{actor}]$
if algorithm is DDPG or TD3 **then**
  Initialize replay buffer with size B
  Initialize target network parameters Θ'
end if
for episode $k=1,\ K$ **do**
  Initialize environment // randomly given road length l
  for $t=1,\ T_E$ **do**
    Get state $s_{t,j}$
    Select $a_{t,j}\sim\pi_{\Theta_{actor}}(s_{t,j})$ // by actor network
    Execute action $a_{t,j}$ and compute reward $r_{t,j}$ and get new state $s_{t+1,j}$
    Store trajectory $<s_{t,j}, a_{t,j}, r_{t,j}, s_{t+1,j}>$
    Update critic network parameter Θ_{critic} using gradient descent step:
$$\Theta_{critic}\leftarrow\Theta_{critic}-\alpha\nabla_{\Theta_{critic}}L$$
    if TD3 and t mod ϕ or DDPG or PPO **then**
      Update actor network parameter Θ_{actor} using gradient ascent step:
$$\Theta_{actor}\leftarrow\Theta_{actor}+\alpha\nabla_{\Theta_{actor}}L$$
      if TD3 or DDPG **then**
        Update target network parameters Θ' using soft update:
$$\Theta'\leftarrow\tau\Theta+(1-\tau)\Theta'$$
      end if
    end if
  end for
end for

알고리즘 1. 자율주행차량을 위한 심층 강화학습 기반 가속도 제어 의사코드
Algorithm 1. Pseudo-code of deep reinforcement learning based acceleration control algorithm for autonomous vehicles

$$\hat{A}_t=\delta_{t,j}+\gamma\delta_{t+1,j}+\cdots+\gamma^{T-t+1}\delta_{T-1,j}\quad where\quad \delta_{t,j}=r_{t,j}+\gamma V_{t+1,j}^{\pi}(s)-V_{t,j}^{\pi}(s) \tag{5}$$

$$V_t^{CLIP}=V_t^{pred}+\min(\max(V_{\theta^{VF}}(s_{t,j})-V_t^{pred}),-\epsilon),\epsilon) \tag{7}$$

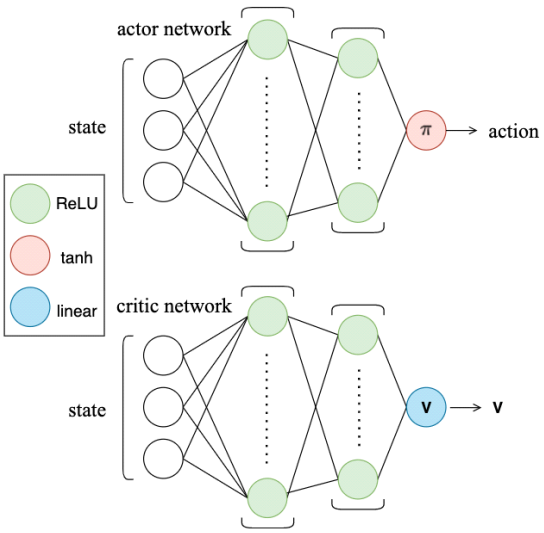


그림 3. PPO 의 심층 신경망 구조
Fig. 3. The neural network architecture of PPO

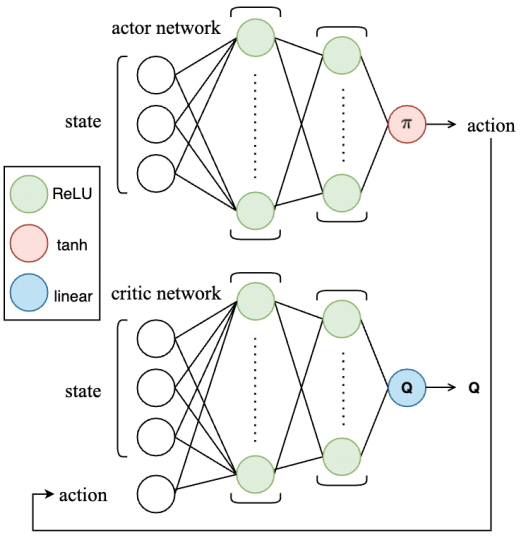


그림 4. DDPG 의 심층 신경망 구조
Fig. 4. The neural network architecture of DDPG

심층 신경망에 의해 이루어진다. PPO 알고리즘의 심층 신경망 구조는 그림 3과 같다. 액터 네트워크의 경우 개체가 관측한 상태 정보 $s_{t,j}$ 를 입력값으로 사용한다. 입력값은 은닉층(hidden layer)을 거쳐 정책 π에서 행동 $a_{t,j}$를 출력한다. 크리틱 네트워크의 경우 동일하게 상태 정보 $s_{t,j}$를 사용하지만 은닉층을 거쳐 가치함수 V^{π}를 출력한다. 각 계층의 활성화 함수 및 은닉층의 각 계층과 노드는 사용자에 의해 결정된다.

4.2 Deep Deterministic Policy Gradient (DDPG)

DDPG 의 네트워크 집합 $\Theta=[\theta^{\mu},\theta^{Q}]$ 역시 액터 네트워크 θ^{μ} 와 크리틱 네트워크 θ^{Q}로 구분할 수 있다. 또한 DDPG 의 경우 Off-policy 기반의 알고리즘으로 목표 네트워크 집합 $\Theta'=[\theta^{\mu'},\theta^{Q}]$를 갖는다. Q-함수를 위한 크리틱 네트워크의 목적함수 $L(\theta^{Q})$ 는 다음과 같다.

$$L(\theta^{Q})=\frac{1}{M}\sum_{t}\left[(Q(s_{t,j},a_{t,j}\mid\theta^{Q})-y_{t})^{2}\right] \tag{8}$$

목표 $y_{t}=r(s_{t,j},a_{t,j})+\gamma Q^{\mu'}(s_{t+1,j},\mu(s_{t+1,j}))$는 보상과 목표 정책 μ' 하에서의 Q-함수 값의 합을 의미한다. 이때 $\mu(s_{t+1,j})=\mathrm{argmax}_{a_{t+1,j}}Q^{\mu}(s_{t+1,j},\mu(s_{t+1,j}))$ 와 같이 행동 $a_{t+1,j}$를 선택할 때 결정론적인 argmax 방법을 사용한다. M은 Batch size를 의미한다. 크리틱 네트워크는 식 (8)의 목적함수가 최소화 되는 방향으로 학습된다.

DDPG 의 정책 최적화를 위한 액터 네트워크의 목적함수는 다음과 같다.

$$L(\theta^{\mu})\approx\frac{1}{M}\sum_{t}Q(s,\mu(s\mid\theta^{\mu})\mid\theta^{Q}) \tag{9}$$

DDPG 는 매번 목표 네트워크를 다음과 같은 방식으로 업데이트한다.

$$\theta^{Q'}\leftarrow\tau\theta^{Q}+(1-\tau)\theta^{Q'} \tag{10}$$

$$\theta^{\pi'}\leftarrow\tau\theta^{\mu}+(1-\tau)\theta^{\mu'} \tag{11}$$

식 (10)의 θ^{Q}는 크리틱 네트워크의 Q-함수를 근사하는 파라미터이며 $\theta^{Q'}$는 목표 Q-함수를 근사하는파라미터이다. 식 (11)의 θ^{μ}는 액터 네트워크의 정책 근사 파라미터이며 $\theta^{\mu'}$는 목표 정책을 근사하는 파라미터이다.

각각의 네트워크의 업데이트 수식에 사용된 τ는 목표 네트워크의 변화율을 조절하는 파라미터이다. $\tau\in[0,1]$ 가 1에 가까울수록 목표 네트워크의 파라미터는 크게 변화하며 0에 가까울수록 적게 변화한다. 이러한 방식을 Soft update 라고 하며 DDPG

는 이를 통해 목표 네트워크가 천천히 변화할 수 있도록 강제한다. 목적함수의 근사는 심층 신경망을 통해 이루어지며 DDPG 의 심층 신경망 구조는 그림 4를 통해 확인할 수 있다.

4.3 Twin Delayed DDPG (TD3)

TD3의 네트워크 집합 $\Theta=[\theta^{\mu},\theta^{Q_1},\theta^{Q_2}]$ 및 목표 네트워크 집합 $\Theta'=[\theta^{\mu'},\theta^{Q_1'},\theta^{Q_2'}]$은 DDPG의 네트워크에서 크리틱 네트워크가 한 개 더 추가된 구성이다. 또한, TD3에서 사용되는 목적함수는 DDPG와 동일한 구성을 가지며 크리틱 네트워크에서 사용되는 식 (8)의 목표 함수 y_t를 구하는 방법만 식 (12)로 변경된다.

이는 DDPG의 과대추정 편향(overestimation bias)을 해결하기 위해 개선된 방법이다. $\min_{i=1,2}Q_{\theta^{Q_i'}}$을 통해 두 네트워크에서 근사한 Q-함수 중 더욱 작은 값을 사용한다. 또한 목표 정책에 평활화(smoothing) 기법을 적용하여 행동 선택 과정에서 클리핑 무작위 잡음(clipped random noise) w를 가한다.

TD3와 DDPG는 네트워크 업데이트 방식에서도 차이를 보인다. DDPG의 경우 정해진 Time step 마다 모든 네트워크가 차례로 학습을 하는 반면 TD3는 지연 업데이트(delayed update) 방식을 사용한다. 이는 크리틱 네트워크보다 액터 네트워크와 목표 네트워크의 업데이트 주기를 늦추는 방식이다. 이를 통해 Q-함수가 안정되어 다른 네트워크에서 발생하는 과대 추정 및 오류의 축적을 방지할 수 있다. 결과적으로 분산이 낮은 값을 추정할 수 있도록 하며 정책의 품질을 보장한다.

TD3의 심층 신경망 구조 역시 전반적으로 DDPG 와 유사하다. 하지만 TD3 의 경우 Twin Q-learning을 적용하였기 때문에 Q-함수 Q_1과 Q_2에 대한 네트워크 파라미터 θ_1과 θ_2가 각각 존재한다는 점을 그림 5를 통해 확인할 수 있다.

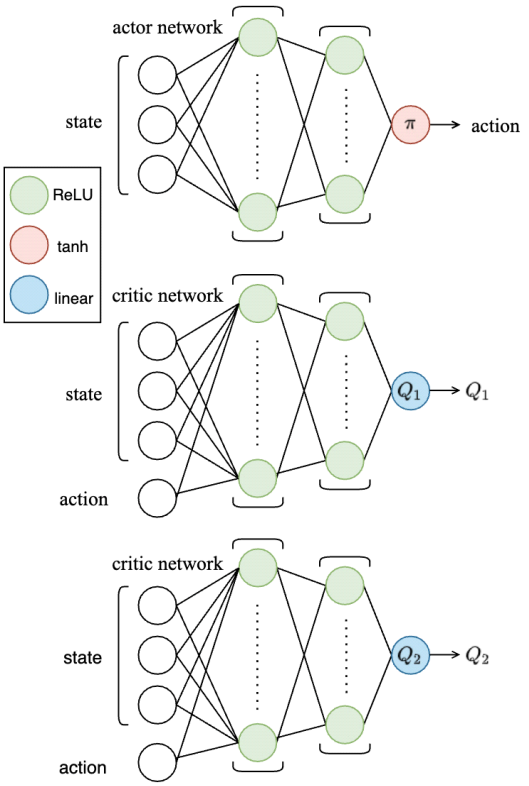


그림 5. TD3 의 심층 신경망 구조
Fig. 5. The neural network architecture of TD3

V. 시뮬레이션을 통한 알고리즘 성능 분석

5.1 시뮬레이션 및 학습 환경 설정

본 연구에서 진행된 모든 시뮬레이션 및 학습은 교통 시뮬레이터 SUMO[30]와 분산 처리 API인 RAY[31]에서 제공하는 심층 강화학습 라이브러리 RLlib[32]을 병합한 프레임워크인 FLOW[33]를 통해 진행하였다.

도로는 그림 1과 같은 원형 구조이다. 시뮬레이션 과정에서 도로의 길이는 $l=260m$으로 고정되어 있다. 반면 자율주행차량 e_j를 학습시키는 과정에서는 특정 도로 길이에 과적합(overfitting)되는 것을 방지하기 위해 모의실험(simulation)이 초기화 될 때마다 $l\in[220m,270m]$에서 임의로 설정한다. 학습 및 성능 평가는 모의실험 환경에서 진행된다. 모의실험은 하나의 시동(warm-up)기간 $T_W=750ts$과 에피소드 $T_E=3000ts$의 두 단계로 구성된다. 즉, 한 번의 에피소드는 $T=T_W+T_E=3750ts$로 구성된다. 시동 기간은 고전적으로 시뮬레이션의 시작 단계에서 발생하는 스타트업 문제(start-up problem)[34]를 해결하기 위해 설정하였다. 시동기간 동안의 데이터는 학습에 포함하지 않으며 본 연구에서

$$y_t=r(s_{t,j},a_{t,j})+\gamma\min_{i=1,2}Q_{\theta^{Q_i'}}(s_{t+1,j},\mu(s_{t+1,j})+w)\quad w\sim clip(N(0,\sigma),-\epsilon,\epsilon) \tag{12}$$

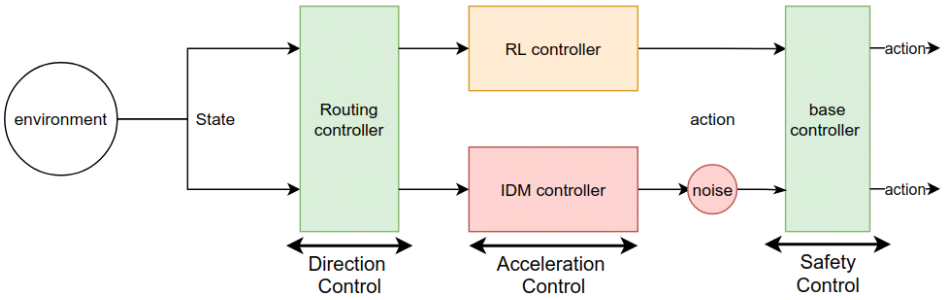


그림 6. 차량의 컨트롤러 구성
Fig. 6. The controller architecture of vehicles

$1ts = 0.1s$ 이다.

도로를 주행하는 차량의 수는 $N=22$이다. 도로 내 자율주행차량의 수는 1대이며, 그림 1의 빨간색 차량이 자율주행차량 e_j이다. 나머지 21대의 차량은 모두 비 자율주행차량이며, 그림 1의 파란색 차량은 자율주행차량이 학습 중 관측할 수 있는 선두 차량 e_{j-1}이다. 흰색 차량은 모두 비 자율주행차량으로, 자율주행 차량이 관측할 수 없는 차량 $e_{i,\, i \notin j,\, j-1}$ 이다. 모든 차량의 크기는 동일하게 5m로 설정하였다. 자율주행차량의 행동은 $A = \{a_{t,j} \mathit{IN} \mathbb{R} \mid -1 \le a_{t,j} \le 1\}$에서 정의된다.

5.2 차량의 컨트롤러 구성

차량의 행동 선택 및 컨트롤러(controller)의 구성은 그림 6과 같다. 모든 차량은 도로 구조에 대한 경로 설정 컨트롤러(routing controller)를 갖고 있어 매 Time step 마다 경로를 계산하여 진행 방향에서 벗어나지 않도록 조절한다. 또한, 모든 차량은 기본 컨트롤러(base controller)를 포함한다. 기본 컨트롤러는 지능형운전보조시스템(advanced driver assistance system)[35]과 같은 안전보조 역할을 담당한다. 예를 들어, 선두차량과의 거리가 안전거리보다 가까워지면 급브레이크를 밟게 되는 등의 행동을 수행한다.

자율주행차량은 RL 컨트롤러를 통해 학습된 정보를 바탕으로 특정 상태에 맞는 행동 $a_{t,j}$ 를 수행한다. 반면, 비 자율주행차량의 경우 Intelligent Driving Model[36] (IDM) 컨트롤러를 사용하여 선두차량과의 안전거리를 유지하며 행동 $a_{t,i}$(단, $i \neq j$)을 결정하도록 설정하였다.

IDM 컨트롤러는 다음의 식을 통해 비 자율주행차량의 가속도 $a_{t,i}$(단, $i \neq j$)를 조절한다.

$$a_{t,i} = 1 - (\frac{v_{t,i}}{v^*})^{\delta} - \left(\frac{g(v_{t,i}, v_{t,i} - v_{t,i-1})}{f(d_{t,i-1} - d_{t,i})}\right)^2 \quad (\text{단},\ i \neq j) \tag{13}$$

이때, v^*는 목표속도, δ는 속도 지수(velocity exponent)를 의미한다. $v_{t,i} < v^*$일 때 $1 - (\frac{v_{t,i}}{v^*})^{\delta} > 0$ 이므로 양의 가속을 결정한다. 목표 속도 $v^* = v_{t,i}$ 의 경우에는 가속을 0으로 하여 등속 운동을, $v^* < v_{t,i}$ 일 때는 음의 가속을 결정하도록 유도하여 차량이 v^* 를 유지할 수 있도록 설계하였다. 식 (13)의 세 번째 항 $\left(\frac{g(v_{t,i}, v_{t,i} - v_{t,i-1})}{f(d_{t,i-1} - d_{t,i})}\right)^2$은 비 자율주행차량 e_i와 e_i의 선두 차량 e_{i-1} 사이의 안전거리를 유지할 수 있도록 보조하는 역할을 한다. 여기서 함수 g는 식 (14), f는 수식 (2)를 따르며 $d^* = 2m$, $t^* = 1s$로 설정하였다. 이때 t^*은 Time headway로 e_i와 e_{i-1}이 동일한 위치에 도달하는데 걸리는 시간의 차이를 의미한다. IDM 컨트롤러로 결정한 $a_{t,i}$(단, $i \neq j$)에 $N(0, 0.2)$의 가우시안 분포(Gaussian distribution)를 갖는 잡음을 더해주어 실제 운전자들의 얘기치 못한 행동 또한 표현 가능하도록 설정하였다[37].

5.3 자율주행 차량에서 사용된 심층 신경망 구조

$$g(v_{t,i}, v_{t,i} - v_{t,i-1}) = d^* + \max\left[0, (v_{t,i} \times t^* + \frac{v_{t,i}\,(v_{t,i} - v_{t,i-1})}{2\sqrt{a_{\max} \times \mid a_{\min} \mid}}\right] \tag{14}$$

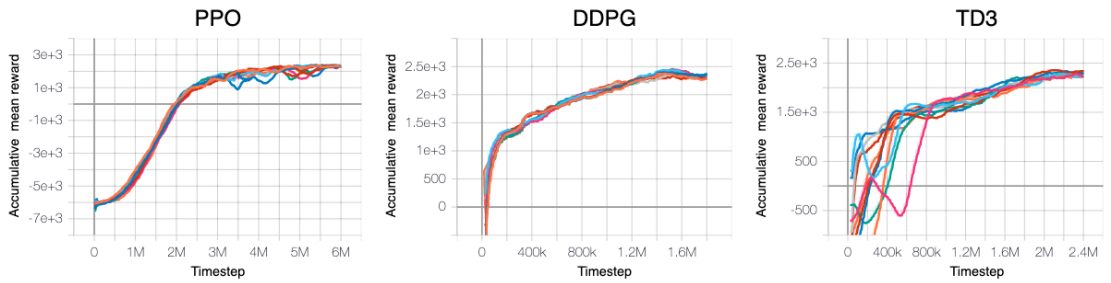


그림 7. 심층 강화학습 알고리즘별 시간에 따른 단일 에피소드의 누적 평균 보상
Fig. 7. The Accumulative mean reward of an episode over time steps

및 학습 설정

본 연구에서 사용된 심층 강화학습 알고리즘 PPO, DDPG, TD3는 모두 액터-크리틱 구조를 가진다. 본 논문에서 사용한 활성화 함수 및 노드와 계층 정보는 Appendix 1 - 표 A1에 제시되어 있다. 신경망의 효율적 학습 과정을 위한 Optimizer는 Adam[38]을 사용하였다.

PPO의 크리틱 네트워크는 가치함수를 근사한다(그림 3, 4, 5). PPO의 모든 네트워크의 입력값으로는 상태 정보를 사용한다. 본 논문에서 사용한 PPO의 심층 신경망 구조는 그림 3에 제시되어 있다. 반면, DDPG와 TD3의 크리틱 네트워크는 Q-함수를 근사한다. 따라서 크리틱 네트워크의 입력값으로 상태 정보와 행동 정보가 함께 사용된다. 본 논문에서 사용한 DDPG, TD3의 심층 신경망 구조는 그림 4, 5에 각각 제시되어 있다.

학습 초기 더 나은 정책을 찾기 위한 탐험(exploration)과정은 PPO의 경우 확률적 행동 추출(stochastic sampling) 방식으로 진행된다. DDPG는 Ornstein-Uhlenbeck noise[39]를 더하는 방식으로, TD3는 Gaussian noise를 더하는 방식으로 구현하였다. 각 알고리즘별 자세한 파라미터 설정은 Appendix I - 표 A3를 통해 확인할 수 있다. TD3에서만 사용된 Policy delay ϕ는 Ⅲ-다에서 언급했던 지연 업데이트를 위해 필요한 파라미터로, 크리틱 네트워크가 2번 업데이트 될 때 액터 네트워크와 목표 네트워크는 1번 업데이트 되도록 설정하였다.

5.4 심층강화학습 알고리즘별 성능 평가

본 논문에서는 세 가지 알고리즘을 이용하여 학습시킨 자율주행차량의 주행성능 평가 및 주행패턴 분석을 진행하였다. 각 알고리즘의 정량적 성능 평가를 위해 단일 에피소드에서의 누적 평균 보상의 변화를 확인하였다. 자율주행차량의 정성적 성능인 주행패턴의 확인을 위해서는 단일 에피소드에서 Time step 변화에 따른 속도 및 가속도의 변화를 확인하였다. 마지막으로 본 논문에서 해결하고자 하였던 원형도로의 Stop-and-go wave 현상 해결 을 평가하기 위해 비 자율주행차량으로만 구성된 네트워크와 비교한다. 이를 위해 비 자율주행차량의 속도 변화 및 Time step에 따른 22대 차량의 속도에 대한 분산을 확인하였다. 성능평가를 위해 10개의 랜덤 시드 번호(random seed number)를 생성하여 랜덤 시드 번호마다 한 대의 차량을 학습시켰다. 알고리즘별 10대의 학습 차량의 결과를 확인할 수 있으며 그림 7의 수렴 값은 Appendix I - 표 A3를 통해 자세하게 확인할 수 있다.

5.4.1 심층 강화학습 학습 과정 분석

심층 강화학습 알고리즘별 학습 과정은 그림 7을 통해 확인할 수 있다. 그림 7은 단일 에피소드에서 얻을 수 있는 누적 보상을 Time step 별로 확인하였다. 세 알고리즘 모두 2200~2350 사이의 값에 수렴하였다. DDPG와 TD3의 경우 PPO에 비해 조금 더 빠르게 수렴하는 모습을 확인할 수 있다. 이는 두 알고리즘이 결정론적 정책을 사용하며 Off-policy 방법을 사용하여 효과적으로 탐험을 하기 때문이라고 해석된다. 이러한 정책의 특성 때문에 수렴 단계에서 TD3와 DDPG의 경우 각 개체가 일정한 값을 유지하는 모습을 확인할 수 있다. TD3의 경우 정책의 업데이트를 지연하며, 두 개의 Q-함수 중 최소값을 선택적으로 사용하여 학습하기 때문에 DDPG 보다는 천천히 학습되는 모습을 확인할 수 있다. 또한, DDPG의 경우 TD3에 비해 과대평가되는 방향의 학습 경향을 확인하였다(Appendix I - 그림 A1).

PPO는 On-policy 방법을 사용하여 목표 정책과

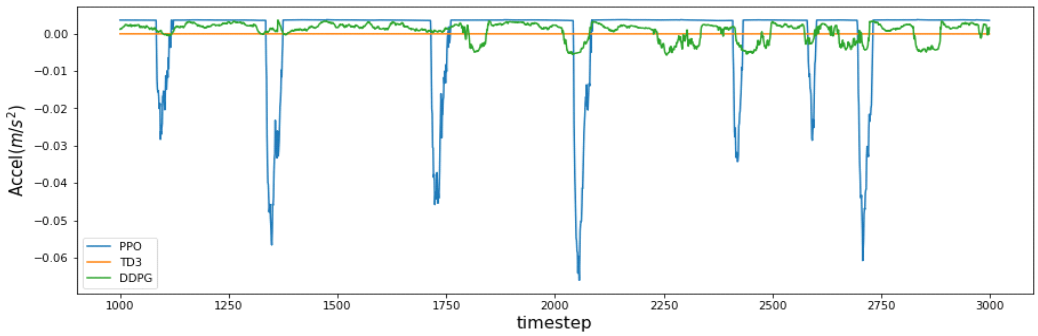


그림 8. 시간에 따른 자율주행차량의 가속도 변화
Fig. 8. The acceleration of autonomous vehicle over time steps

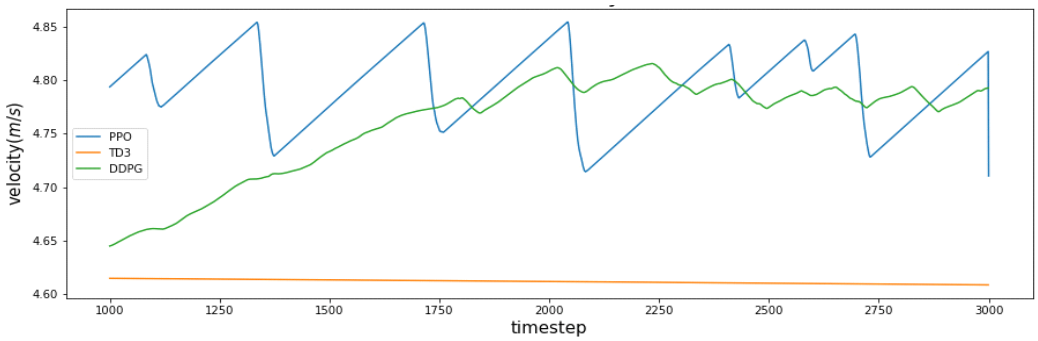


그림 9. 시간에 따른 자율주행차량의 속도 변화
Fig. 9. The velocity of autonomous vehicle over time steps

행동 정책의 구분이 없기 때문에 다른 두 알고리즘에 비해 수렴하기 위해 많은 Time step을 필요로 한다. 또한, PPO의 정책 학습 방식은 분산이 최적화된 정책 분포에서 행동을 선택하는 확률적 정책이다. 이러한 차이 때문에 특정 개체에서 수렴 궤도에 오른 이후에도 곡선을 이탈하는 지점을 찾아볼 수 있다.

5.4.2 심층 강화학습 알고리즘별 자율주행차량의 주행 분석

각 알고리즘 별 자율주행 차량의 주행 성능 및 패턴을 분석한다. 그림 8은 단일 에피소드에서 시간에 따른 자율주행차의 가속도 변화를 나타낸 그래프이며, 그림 9는 시간에 따른 자율주행차의 속도 변화를 나타낸 그래프이다. 그림 8을 통해 가속도의 변화를 살펴보면, PPO로 학습된 차량의 경우 양의 가속도를 유지한 채 등가속도 운동을 하다가 순간적인 음의 가속도로 제동을 하는 모습을 보인다. DDPG로 학습된 차량의 경우 역시 가속 운동을 하며 제동을 하지만 그 정도가 PPO에 비해서는 약한 모습을 보여준다. 이러한 제동은 선두 차량과의 상대적인 거리가 가까워졌을 때 안전거리를 확보하기 위함이다. 반면 TD3로 학습된 차량의 가속도는 주행 내내 0에 수렴하여 등가속도 운동을 하는 모습을 보여주었다. 이러한 주행 패턴의 차이는 그림 9를 통해 자세하게 확인할 수 있다. PPO로 학습된 차량의 속도는 시간의 변화에 따라 계속해서 감·가속하는 모습을 보였다. DDPG로 학습된 차량의 경우 PPO와 비교하여 조금 더 자연스럽게 가속 조절을 한다고 판단하였다. 두 알고리즘으로 학습된 차량의 경우 가속 주행을 지향하기 때문에 약 $4.75m/s \sim 4.85m/s$ 사이의 속도로 지속적으로 변화를 시키며 주행한다. 반면 TD3로 학습된 차량의 시간에 따른 속도 변화 곡선은 $4.62m/s$ 정도의 속도로 등속주행에 가깝게 주행한다는 것을 확인 할 수 있다.

이러한 주행 별 차이는 알고리즘의 특징을 통해 유추할 수 있다. PPO의 경우 다른 두 알고리즘과 달리 확률론적 정책을 사용한다. 즉, 특정 상태에서

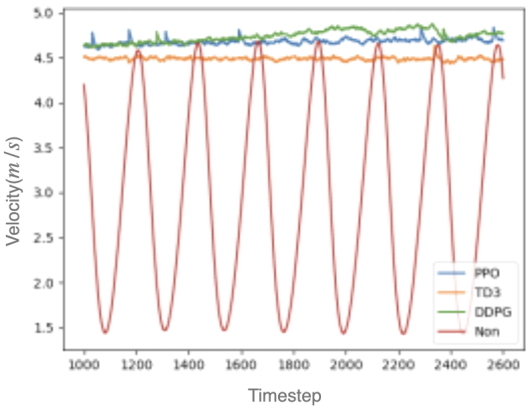


그림 10. 네트워크 별 시간에 따른 비 자율주행차량의 속도 변화
Fig. 10. The velocity of non-autonomous vehicles over time steps

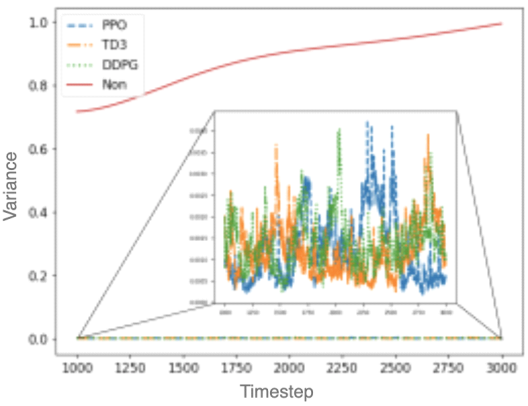


그림 11. 네트워크 별 모든 차량의 시간에 따른 속도의 분산
Fig. 11. The variance of velocities of all vehicles over time steps

취할 수 있는 행동의 분포인 정책에서 샘플링을 통해 행동을 수행한다. 따라서 동일한 상태 정보가 주어진다 할지라도 다른 행동을 수행할 수 있는 것이다. 반면 DDPG와 TD3의 경우는 결정론적 정책을 사용한다. 두 알고리즘에서 결과의 차이는 IV-3에서 언급하였듯 DDPG의 과대추정 및 TD3의 과소추정의 차이로 유추할 수 있다. 두 알고리즘의 차이는 Appendix I - 그림 A1을 통해 단편적으로 확인할 수 있다. 이러한 Q-함수 값의 경향성은 간접적으로 정책에 영향을 미친다. 결과적으로 DDPG는 정책의 변동성이 커지게 되며 TD3는 보다 안정된 속도로 학습이 된다. 이러한 학습 방식의 차이가 자율주행차량의 주행 방식의 차이를 나타내었다고 추론할 수 있다.

5.4.3 자율주행차량의 stop-and-go wave 현상 해결 능력 평가

본 논문에서 궁극적으로 해결하고자 하는 Stop-and-go wave 문제가 해결되었는지를 확인하기 위해 자율주행차량이 도로 내 존재하는 경우와 비 자율주행차량만 도로에 존재하는 경우의 도로 내 속도를 비교한다. 정량적 성능평가를 위하여 각 도로에서 무작위로 선택한 비 자율주행 차량 1대의 속도 변화 및 전체 차량 속도의 분산 값을 확인하였다. 속도 변화 그래프의 진폭을 확인하여 Stop-and-go wave 현상의 유무를 확인할 수 있고, 이를 정량적으로 측정하기 위하여 전체 차량 속도의 분산 값을 이용하였다. 분산 값이 작으면 도로 내 차량들의 속도가 비슷하게 운동하여 Stop-and-go wave 현상이 해결됨을 의미한다. 그림 10은 각 네트워크에서 시간에 따른 비 자율주행차량 한 대의 속도 변화를 나타낸 그래프이다. PPO, TD3, 그리고 DDPG 가 포함된 도로에서의 비 자율주행차량의 속도 변화에서는 큰 진폭을 보이지 않는 결과를 확인할 수 있다. 반면 비 자율주행차량만이 존재하는 도로의 경우 Stop-and-go wave 현상으로 인해 매우 큰 진폭을 확인할 수 있다. 그림 11은 시간에 따른 모든 차량의 속도의 분산을 나타낸 그래프다. 분산이 크다는 것은 구성 차량의 속도가 일정하지 않다는 것이다. 즉, 계속된 속도 변화로 인해 Stop-and-go wave 현상이 해소되지 않았음을 의미한다. 비 자율주행차량으로만 구성된 네트워크의 경우 큰 분산을 보이며 자율주행차량이 존재하는 경우에는 0에 근사할 수준의 분산을 보였다. 각 알고리즘 별 분산의 자세한 경향성 역시 그림 11을 통해 확인할 수 있다. 또한 표 1을 통해 각 네트워크에서의 평균 속도를 확인할 수 있다. 각각의 알고리즘으로 학습된 자율주행차량이 포함된 네트워크는 그렇지 않은 네트워크에 비해 각각 60%(PPO), 59%(DDPG), 그리고 55%(TD3)의 평균속도 증가를 보였다.

표 1. 각 알고리즘별 네트워크의 평균 속도 및 분산
Table 1. Mean and variance of velocities for algorithms

velocity(m/s)	Non	PPO	DDPG	TD3
mean	2.97	4.75	4.72	4.61
variance	0.88	0.001	0.001	0.001

Ⅵ. 결 론

본 연구에서는 심층 강화학습 알고리즘을 통해 자율주행차량을 학습시켜 원형도로에서 발생하는 Stop-and-go wave 현상 해결 및 알고리즘별 주행 패턴을 분석하였다. 본 논문에서 제안한 MDP 모델로 학습시킨 자율주행차량이 존재하는 네트워크의 경우 비 자율주행차량만으로 구성된 네트워크와 비교하였을 때 Stop-and-go wave 현상을 효과적으로 제어하였다. 도로의 평균 속도 역시 자율주행차량이 주행하는 경우 55% 이상의 높은 증가율을 보여주었다. 각 알고리즘을 통해 학습시킨 차량의 성능차이 역시 확인할 수 있었다. 수렴한 상태의 성능지표를 살펴보면 전반적인 지표에서 TD3로 학습시킨 경우 가장 안정적인 모습을 확인할 수 있었다. 정성적 결과를 통해 주행패턴 비교를 수행한 결과 PPO 및 DDPG로 학습시킨 차량의 경우 가속주행을 지향하며 제동을 사용하는 모습을 보였다. 반면 TD3로 학습시킨 차량의 경우에는 등속주행을 지향하는 모습을 확인할 수 있었다. 이러한 주행패턴의 차이로 PPO 및 DDPG의 네트워크 평균 속도는 약 $4.70m/s$, TD3는 약 $4.60m/s$의 값을 보였다.

References

[1] SAE INTERNATIONAL, *Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles* (2018), Retrieved Mar. 18, 2021, from https://www.sae.org/standards/content/j3016_201806/

[2] J. Baek, *Domestic and international development status of self-driving cars* (2020), Retrieved Mar. 30, 2021, https://m.kdb.co.kr/MHFLMN00N00.act

[3] Ministry of Land, Infrastructure and Transport, *Establishment of the world's first partial self-driving car (level 3) safety standards* (2020), Retrieved Mar. 27, 2021, http://www.molit.go.kr/USR/NEWS/m_71/dtl.jsp?lcmspage=1&id=95083365

[4] J. Park and H. Kim, *Autonomous driving system* (2019), KISTEP, Retrieved Mar. 27, 2021, https://www.kistep.re.kr/c3/sub8.jsp

[5] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda, "A survey of autonomous driving: Common practices and emerging technologies," in *IEEE Access*, vol. 8, pp. 58443-58469, 2020.

[5] D. González, J. Pérez, V. Milanés, and F. Nashashibi, "A review of motion planning techniques for automated vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 4, pp. 1135-1145, 2016.

[6] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, "A survey of deep learning techniques for autonomous driving," *J. Field Robot.*, vol. 37, no. 3, pp. 362-386, 2019.

[7] W. Schwarting, J. Alonso-Mora, and D. Rus, "Planning and decision-making for autonomous vehicles," *Annu. Rev. Control Robot. Auto. Syst.*, vol. 1, no. 1, pp. 187-210, 2018.

[8] J. Chen, B. Yuan, and M. Tomizuka, "Model-free deep reinforcement learning for urban autonomous driving," *ITSC*, pp. 2765-2771, Auckland, New Zealand, Oct. 2019.

[9] V. Mnih, K. Kavukcuoglu, D. Silver, et al., "Playing atari with deep reinforcement learning," in *NIPS Deep Learning Wkshp.*, 2013.

[10] J. Wang, R. Chai, and X. Xue, "The effects of stop-and-go wave on the immediate follower and change in driver characteristics," *Procedia Eng.*, vol. 137, pp. 289-298, 2016.

[11] Y. Sugiyama, M. Fukui, M. Kikuchi, et al., "Traffic jams without bottlenecks－experimental evidence for the physical mechanism of the formation of a jam," *New J. Phys.*, vol. 10, no. 3, p. 033001, 2008.

[12] J. Schulman, F. Wolski, P. Dhariwal, et al., "Proximal policy optimization algorithms," arXiv preprint arXiv:1707.06347, 2017.

[13] T. Lillicrap, J. Hunt, A. Pritzel, et al., "Continuous control with deep reinforcement learning," *ICLR*, San Juan, Puerto Rico, May 2016.

[14] S. Fujimoto, H. van Hoof, and D. Meger, "Addressing function approximation error in

actor-critic methods," *ICML*, vol. 80, pp. 1587-1596, 2018.

[15] D. Silver, J. Schrittwieser, K. Simonyan, et al., "Mastering the game of go without human knowledge," *Nature*, vol. 550, pp. 354-359, 2017.

[16] J. Kover, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *Int. J. Robot Res.*, vol. 32, no. 11, pp. 1238-1274, 2013.

[17] A. Folkers, M. Rick, and C. Büskens, "Controlling an autonomous vehicle with deep reinforcement learning," *IEEE IV*, pp. 2025-2031, Paris, France, Jun. 2019.

[18] H. E. Tseng and D. Filev, "Autonomous highway driving using deep reinforcement learning," *SMC 2019*, pp. 2326-2331, Bary, Italy, Oct. 2019.

[19] Z. Zhou, X. Li, and R. N. Zare, "Optimizing chemical reactions with deep reinforcement learning," *ACS Cent. Sci.*, vol. 3, no. 12, pp. 1337-1344, 2017.

[20] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd Ed., The MIT Press, 2018.

[21] R. S. Sutton, D. McAllester, S. Singh, et al., "Policy gradient methods for reinforcement learning with function approximation," *NIPS*, pp. 1057-1063, Massachusetts, USA, 1999.

[22] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Mach. Learn.*, vol. 8, no. 3-4, pp. 229-256, 1992.

[23] P. Varaiya, "Smart cars on smart roads: Problems of control," *IEEE Trans. Automat. Contr.*, vol. 38, no. 2, pp. 195-207, 1993.

[24] M. Vitelli and A. Nayebi, "CARMA: A deep reinforcement learning approach to autonomous driving," *Stanford Project*, pp. 1-8, 2016.

[25] S. Wang, D. Jia, and X. Weng, "Deep reinforcement learning for autonomous driving," arXiv preprint arXiv:1811.11329, 2018.

[26] V. Milanés and S. E. Shladover, "Modeling cooperative and autonomous adaptive cruise control dynamic response using experimental data," *Trans. Res. C Emerg. Technol.*, vol. 48, pp. 285-300, 2014.

[27] R. E. Stern, S. Cui, M. L. D. Monache, et al., "Dissipation of stop-and-go waves via control of autonomous vehicles: Field experiments," *Trans. Res. C Emerg. Technol.*, vol. 89, pp. 205-211, 2018.

[28] Y. Wang, E. Sarkar, W. Li, M. Maniatakos, and S. E. Jabari, "Stop-and-Go: Exploring backdoor attacks on deep reinforcement learning-based traffic congestion control systems," arXiv preprint arXiv:2003.07859, 2020.

[29] A. R. Kreidieh, C. Wu, and A. M. Bayen, "Dissipating stop-and-go waves in closed and open networks via deep reinforcement learning," *ITSC*, pp. 1475-1480, Maui, HI, Nov. 2018.

[30] P. A. Lopez, M. Behrisch, L. Bieker-walz, et al., "Microscopic Traffic Simulation using SUMO," *ITSC*, pp. 2575-2582, Maui, HI, Nov. 2018.

[31] P. Moritz, R. Nishihara, S. Wang, et al., "Ray: A distributed framework for emerging AI applications," *OSDI*, pp. 516-577, Carlsbad, CA, Oct. 2018.

[32] E. Liang, R. Liaw, P. Moritz, et al., "RLlib: abstractions for distributed reinforcement learning," *ICML*, vol. 80, pp. 3053-3062, 2018.

[33] C. Wu, A. Kreidieh, K. Parvate, et al., "Flow: Architecture and benchmarking for reinforcement learning in traffic control," arXiv preprint arXiv:1710.05465, 2017.

[34] J. R. Wilson and A. A. B. Pritsker, "Evaluation of startup policies in simulation experiments," *SIMULATION*, vol. 31, no. 3, pp. 79-89, 1978.

[35] A. Ziebinski, R. Cupek, D. Grzechca, et al., "Review of advanced driver assistance systems (ADAS)," *AIP Conf. Proc.*, vol. 1906, no. 1, p. 120002, 2017.

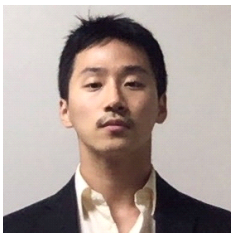
[36] M. Treiber, A. Hennecke, and D. Helbing, "Congested traffic states in empirical observations and microscopic simulations," *Phys. Rev. E*, vol. 62, no. 2, pp. 1805-1824, 2000.

[37] D. Lee, S. W. Kim, and M. Kwon, “Deep reinforcement learning based energy-efficient control for autonomous vehicles,” *JCCI*, Busan, Korea, Apr. 2021.

[38] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” arXiv preprint arXiv:1412.6980, 2014.

[39] G. E. Uhlenbeck and L. S. Ornstein, “On the theory of the brownian motion,” *Physical Rev.*, vol. 36, no. 5, p. 823, 1930.

이 동 수 (Dongsu Lee)



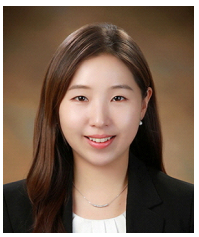
2016년 3월~현재 : 숭실대학교 의생명시스템학부 빅데이터컴퓨팅융합전공

2020년 12월~현재 : 숭실대학교 Brain and Machine Intelligence Lab. 학부생 연구원

<관심분야> 강화학습, 계산신경과학, 자율주행

[ORCID:0000-0002-9238-4106]

권 민 혜 (Minhae Kwon)



2011년 8월 : 이화여자대학교 전자정보통신공학과 학사

2013년 8월 : 이화여자대학교 전자공학과 석사

2017년 8월 : 이화여자대학교 전자전기공학과 박사

2017년 9월~2018년 8월 : 이화여자대학교 전자전기공학과 박사 후 연구원

2018년 9월~2020년 2월 : 미국 Rice University, Electrical and Computer Engineering, Postdoctoral Researcher

2018년 9월~2020년 2월 : 미국 Baylor College of Medicine, Center for Neuroscience and Artificial Intelligence, Postdoctoral Researcher

2020년 3월~현재 : 숭실대학교 전자정보공학부 IT융합전공 조교수

<관심분야> 강화학습, 계산신경과학, 자율주행, 모바일 네트워크, 이상탐지기술

[ORCID:0000-0002-8807-3719]

Appendix

표 A1. 신경망의 노드와 계층 정보
Table A1. Information of layers and nodes for neural network

	input node(actor/ critic)	hidden layer number	hidden layer node	output node(policy /value function)
PPO	3/3	2	[256, 256]	2/1
DDPG	3/4	2	[64, 64]	1/1
TD3	3/4	2	[64, 64]	1/1

표 A2. 연구 환경 시스템 사양
Table A2. System specification of the research

CPU	AMD Ryzen 9 3950X 16-core
GPU	GeForce RTX 2080 Ti
RAM	128GB
SSD	1T

표 A3. 알고리즘별 하이퍼파라미터 설정
Table A3. Hyperparameter according to the algorithm

Hyperparameter	PPO	DDPG	TD3
α_{actor}	5×10^{-7}	10^{-4}	10^{-7}
α_{critic}	5×10^{-7}	10^{-4}	10^{-6}
M	3000	128	64
γ	0.99	0.99	0.99
τ	null	10^{-3}	10^{-3}
B	null	3×10^{5}	3×10^{5}
σ	null	null	0.2
ϕ	null	null	2
ϵ	0.2	null	0.5

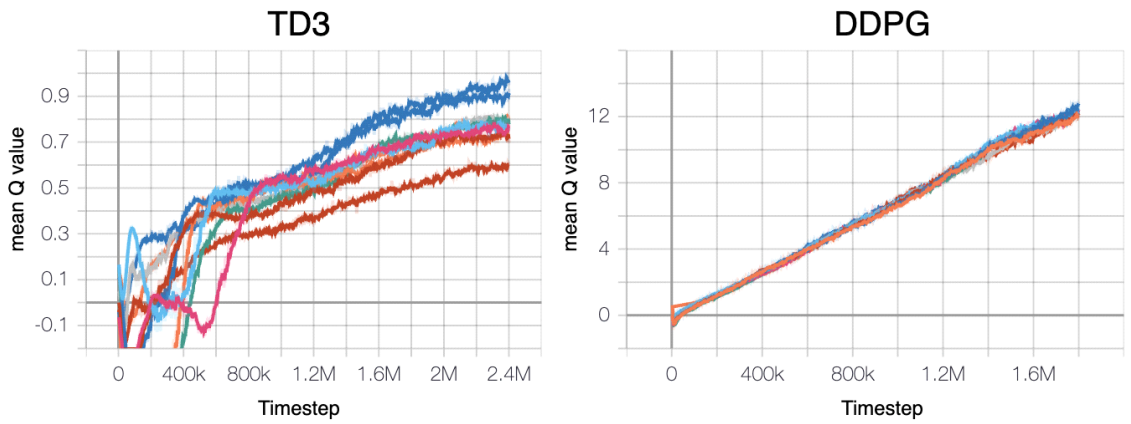


그림 A1. Q-함수 학습 곡선
Fig. A1. Learning curve of Q-function

표 A4. 기호 정리
Table A4. Summary of Notations

Notation	Description	Notation	Description
S	state space	A	action space
s	state	a	action (acceleration)
r	reward	γ	discount factor
π	stochastic policy	μ	deterministic policy
V	state value function	Q	action value function
E	set of vehicles on the roads	e_j	an autonomous vehicle
e_{j-1}	a header vehicle of e_j	N	number of vehicle on ring roads
D	set of possible	d	position of vehicle
d^*	a minimum desired distance	t^*	desired time headway
l	road length	v	velocity of vehicle
δ	velocity exponent	v^*	desired velocity
Θ	network parameters	Θ'	target network parameters
ϵ	clipped parameter	M	batch size
τ	soft update parameter	α	learning rate
B	replay buffer size	σ	stddev for smoothing noise
ϕ	policy delay for each update of the Q	t	time step
K	number of total episodes	T_E	time steps of an episode
T_W	time steps of warm-up stage		