Journal of KIIT. Vol. 22, No. 11, pp. 29-38, Nov. 30, 2024. pISSN 1598-8619, eISSN 2093-7571
http://dx.doi.org/10.14801/jkiit.2024.22.11.29

다중 험지 환경 자율주행을 위한 동기식 어드밴티지 액터-크리틱 모델

장승균*, 정동원**[1], 온병원**[2], 정현준**[3]

Synchronous Advantage Actor-Critic Model for Autonomous Driving in Multiple Rough Terrain Environments

Seungkyun Jang*, Dongwon Jeong**[1], Byung-Won On**[2], and Hyunjun Jung**[3]

이 연구는 정부(과학기술정보통신부)의 재원으로 한국 연구재단의 지원을 받아 수행되고 있습니다
(No. NRF-2022R1G1A1008493)

요 약

오늘날 AI 발전함에 따라 사람들의 노동력을 보완해 주는 자율주행로봇이 개발되었다. 하지만 험지 환경의 경우 복잡한 환경과 변인 요소로 인해 단일 시뮬레이션으로 자율주행로봇의 학습에 한계가 있다. 이를 보완하기 위해 이 논문에서는 sA2C 알고리즘을 사용하여 다중환경에서 하나의 모델을 학습시키는 강화학습 방법을 제안하며, 다중 험지 환경을 구축하여 다양한 변인 요소의 학습을 자율주행로봇에 적용하는 실험을 진행하였다. 추가로 기존 A3C 알고리즘과 제안한 sA2C 알고리즘의 성능 비교를 진행하였으며, 실험 결과 sA2C 알고리즘이 A3C 알고리즘보다 actor loss 값이 48.28%, critic Loss는 44.63%로 더 낮았으며, 평균 보상 값은 43.09%로 더 높았다.

Abstract

Today, with the advancement of AI, self-driving robots have been developed to complement human labor. However, in the case of rough terrain, there are limits to the learning of autonomous robots through a single simulation due to the complex environment and variable factors. To complement this, this paper proposes a reinforcement learning method that uses the sA2C algorithm to learn one model in multiple environments, and conducted an experiment to construct a multiple rough terrain environment and apply learning of various variable elements to an autonomous robot. . Additionally, a performance comparison between the existing A3C algorithm and the proposed sA2C algorithm was conducted, and the experimental results showed that the sA2C algorithm had a lower actor loss value of 48.28%, a critic Loss value of 44.63%, and a higher average reward value of 43.09% than the A3C algorithm.

Keywords

reinforcement learning, autonomous-driving, multi-agents, unity, synchronous advantage actor-critic

* 국립군산대학교 컴퓨터소프트웨어학부 학사과정
- ORCID: https://orcid.org/0009-0007-8658-1174
** 국립군산대학교 컴퓨터소프트웨어학부 교수(**[3] 교신저자)
- ORCID[1]: https://orcid.org/0000-0001-9881-5336
- ORCID[2]: https://orcid.org/0000-0001-6929-3188
- ORCID[3]: https://orcid.org/0000-0002-6717-1395

· Received: Aug. 26, 2024, Revised: Oct. 14, 2024, Accepted: Oct. 17, 2024
· Corresponding Author: Hyunjun Jung
School of Computer Software at Kunsan National University, 558, Daehak-ro, Kunsan-si, Jeollabuk-do, Republic of Korea
Tel.: +82-63-469-8917, Email: junghj85@kunsan.ac.kr

Ⅰ. 서 론

오늘날 AI의 등장으로 사람들의 편의성을 높여주는 자율주행 로봇이나 IoT 시스템 등 다양한 기술들이 등장하였다[1][2]. 이 중에서도 자율주행로봇은 사람의 개입 없이 자체 센서와 학습된 데이터를 이용하여 특정 위치까지 이동하여 사람의 일을 도와주는 로봇이다. 예를 들어, 일상생활에서의 자율주행로봇은 식당에서 음식을 손님에게 전달하거나 대형 건물에서 사람들에게 길을 안내하는 로봇들이 개발되었으며, 이 외에도 군사용 탐지 로봇이나 농업용 수확물 운반 로봇 등 특정 환경에서만 사용되는 자율주행로봇도 개발되었다. 이러한 기능은 사람들에게 부족한 노동력을 보충하거나 위험한 환경에서 사람의 안전을 지키는 역할을 한다. 최근 복잡한 환경에서 자율주행 성능을 향상시키기 위한 연구가 강화학습을 활용해 진행되고 있다[3]-[5]. 먼저 강화학습이란 학습의 주체인 에이전트(Agent)가 특정 목표를 학습하기 위해 주어진 환경(Environment)에서 본인의 상태에 따른 행동을 진행한다. 그 후, 환경은 행동에 대한 보상(Reward)을 에이전트에게 전달하게 되고 에이전트는 얻은 보상을 통해 이전의 행동이 학습 목표와 올바른 행동인지 판단하여 학습 목표에 맞게 모델을 학습시킨다. 이러한 학습 방법을 통해 에이전트는 다양한 상태에서 여러 가지 행동을 진행하고 그중에서 가장 목표에 적합한 최적의 행동을 위주로 학습하게 된다. 이 방법을 활용하여 시뮬레이션 환경을 실제 환경과 유사하게 구현하여 시뮬레이션으로 학습한 모델을 실제 환경에 접목하는 강화학습 방법 자율주행 연구가 진행되었다[6][7]. 시뮬레이션 환경으로 강화학습으로 학습한 자율주행 모델은 실제 환경에서도 학습한 내용과 유사하게 행동을 진행하였다. 하지만 시뮬레이션 환경은 실제 환경에 비해 일반화되어 있는 경우가 많으며 특히 험지 환경의 경우, 하나의 시뮬레이션이 실제 환경의 변인 요소를 다 담지 못한다는 한계가 있다. 이처럼 실제 환경은 시뮬레이션 환경과 달리 다양한 형태와 자율주행에 영향을 미칠 수 있는 변인 요소들이 포함된다. 실제 환경을 최대한 반영하기 위해서 추가 연구 방법으로 멀티에이전트 강화학습 방법을 사용한 다중환경 자율주행 연구가 진행되었다[8]. 대표적인 멀티에이전트 알고리즘인 A3C 알고리즘은 여러 개의 에이전트가 주어진 환경에서 각 에이전트가 네트워크를 병렬 처리하여 하나의 공유 모델에 학습하는 알고리즘이다[9]. 이러한 데이터를 하나의 모델에 전달하여 모델이 다양한 환경에서 학습할 수 있다. 하지만 A3C 알고리즘의 문제점으로 다중환경 중 학습이 되지 않는 환경이 존재할 때, 그 환경의 학습한 데이터도 최종 모델에 전달하여 모델의 성능을 떨어트리는 문제점이 있다. 이 논문에서는 A3C 알고리즘의 문제점을 보완하기 위해 sA2C 알고리즘을 제안하며, 이 방법을 활용하여 sA2C 알고리즘 학습 방법은 기존 A3C 알고리즘같이 각 환경에서 학습한 데이터를 최종 모델에 전달하여 업데이트하는 방식과 같다. 하지만 sA2C 알고리즘은 데이터를 최종 모델에 전달하기 전에 각 환경에서 학습 성능에 따라 중요도를 구하고 학습 환경에 적용하는 비율을 조절하여 모델에 반영한다. 이로 인해, 학습 성능이 낮은 환경에서는 낮은 중요도 값을 가지며, 이는 최종 모델에 낮게 반영하여 모델의 성능 저하를 보완해 주는 역할을 한다. 추가로 sA2C 알고리즘을 성능을 확인하기 위해, 변인 요소가 unity 포함된 다중 험지 환경을 unity 가상환경을 통해 학습 환경을 구축하여 다중 험지 환경에서 로봇이 특정 목표 지점까지 이동하는 자율주행 학습을 통해 기존 A3C 알고리즘과 sA2C 알고리즘의 성능 비교실험을 진행하였다.

이 논문의 2장에서는 A2C 알고리즘과 A3C 알고리즘을 소개하여 sA2C 알고리즘의 이전 연구에 관해 설명하고, 추가로 A3C 알고리즘을 활용하여 다중환경 자율주행을 학습한 연구를 소개한다. 3장에서는 sA2C 알고리즘에 대한 설명과 sA2C 알고리즘을 실험하기 위한 험지 환경을 unity 가상환경으로 구축하고 소개한다. 4장에서는 unity 가상환경을 통해 기존 A3C 알고리즘과 제안한 sA2C 알고리즘을 성능 비교를 진행하여 각 알고리즘의 loss 값과 평균 보상 값을 측정하였다. 5장에서는 최종적으로 전체적인 논문에 대한 요약과 결론을 논하며 향후 과제에 관해 설명한다.

II. 관련 연구

2.1 A2C 알고리즘

A2C(Advantage Actor-Critic) 알고리즘은 현재 상태에서 최적의 행동, 즉 최적의 정책을 찾기 위한 Actor 네트워크와 행동에 대한 보상을 통해 미래 보상의 기댓값을 구하는 critic network로 총 2개의 구성된 강화학습 알고리즘이다. 이 방법은 actor 네트워크에서 구한 최적의 정책을 통해 critic 네트워크의 기댓값을 높여준다. 또한 critic 네트워크에서 구한 기댓값을 통해 현재 상태에 대한 가치를 더욱 정확하게 예측함으로써, Actor 네트워크에 최적의 정책을 선택하는 것에 도움을 준다. 추가로 A2C 알고리즘은 advantage라는 특정 행동의 기댓값과 상태에 대한 평균 기댓값의 차이를 통해 actor 네트워크가 불안전한 결과를 가져오는 것을 방지한다. 이처럼 A2C 알고리즘은 험지 환경같이 행동마다 복잡한 상태를 가지는 환경에서 최적의 정책을 찾는 강화학습 알고리즘에 적합한 방법이다.

2.2 A3C 알고리즘

A3C 알고리즘은 기존 A2C 알고리즘을 발전시켜 하나의 모델이 아닌 다중 모델을 하나의 네트워크에 학습시키는 멀티에이전트 강화학습 알고리즘이다. A3C 알고리즘을 그림 1을 통해 자세히 설명하자면, 각 환경에는 학습 모델인 local network가 존재한다. 이 local network들은 각 환경에서 독립적인 학습을 진행하게 되고 global network에 전달한다. 이후 local network에서 받은 파라미터 값을 global network에 반영하여 모델을 업데이트하고 새로운 파라미터 값을 구하면, 이전에 전달받은 local network에 전달하여 global network와 동일한 파라미터 값을 가지게 한다. 이 과정을 반복하면, global network에는 모든 local network의 학습한 파라미터를 전달받아 local network의 전체 내용을 학습하여 최종적으로 모든 환경을 학습한 하나의 모델이 만들어지게 된다. 하지만 이 모델의 경우 하나의 local network가 목표에 맞지 않는 학습이 진행될 때도 global network에 파라미터를 그대로 전달된다는 문제점이 있다.

2.3 A3C 알고리즘을 활용한 다중환경 자율주행

A3C 알고리즘을 이용하여 다중환경에서 자율주행 강화학습을 진행한 연구가 진행되었다[10]. 위 연구에서는 도시의 다양한 형태의 도로에서 자율주행 자동차가 이동할 때 특정 위치까지 장애물이나 사람들을 피하고 자동차의 차선을 벗어나지 않게 도와주는 학습을 진행하였다. 학습된 차량은 episode가 진행되면서 특정 목표에 도달하거나 충돌하기 전까지 차량을 움직이면서 장애물과 자신의 위치를 파악한다.

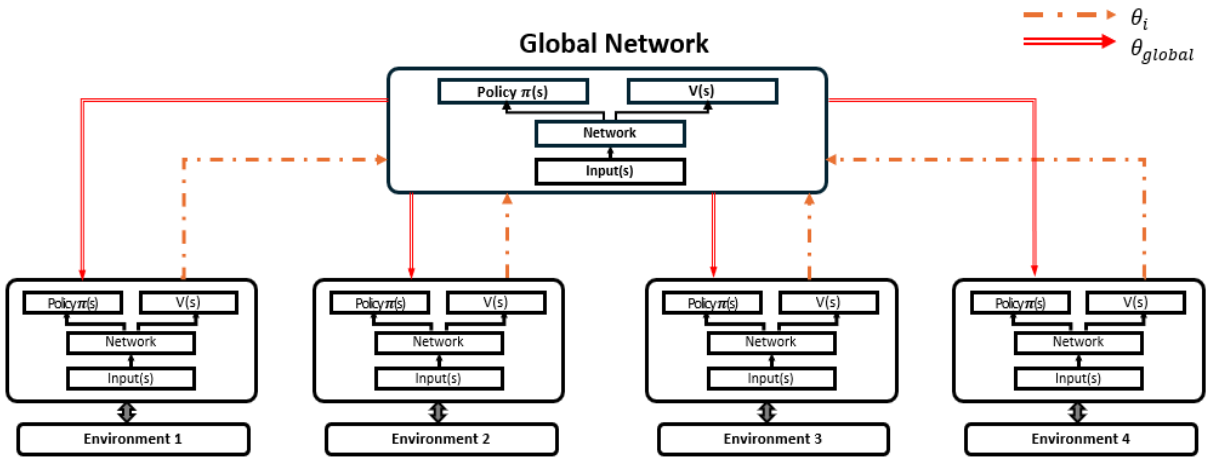

그림 1. A3C 알고리즘 아키텍처
Fig. 1. A3C algorithm architecture

이후 경로에 문제점을 탐색하여 이동에 제한이 생기지 않도록 엔진의 속도나 방향을 조절하여 차량이 올바른 경로로 갈 수 있게 조절을 한다. 올바른 학습을 진행하기 위해 환경은 로봇에게 장애물을 회피할 때 양의 보상을 주거나 목표 지점까지 이동한 거리에 따라 보상을 주는 등 다양한 형태로 보상 함수를 설정하였다. 추가로 A3C 알고리즘의 성능을 평가하기 위해 PPO, IMPALA 알고리즘 등 다양한 강화학습 알고리즘으로 비교실험을 진행하였다[11][12]. 실험 결과로 A3C 모델이 단일 환경과 다중환경에서 다른 알고리즘에 비해 높은 보상을 보여줬으며 A3C 알고리즘이 자율주행 모델에 적용할 수 있다는 가능성을 보여주었다.

III. 다중 험지 환경 자율주행을 위한 동기식 어드밴티지 액터-크리틱 모델

3.1 다중 험지 환경을 위한 sA2C 알고리즘

sA2C(synchronous Advantage Actor-Critic)알고리즘은 A3C(Aynchronous Advantage Actor-Critic) 알고리즘에서 변형된 강화학습 알고리즘이다.

sA2C 알고리즘을 표 1의 수도 코드와 그림 3의 아키텍처를 통해 자세히 설명하자면, 기본적으로 local network에서 학습이 진행된 내용을 global network에 전달하는 것은 기존 A3C 알고리즘과 동일하다. 하지만 sA2C 알고리즘은 local network에서 global network에 전달할 때, local network의 파라미터를 환경의 중요도에 따라 전달되는 파라미터가 달라진다는 점이 기존 A3C 알고리즘에 차이점을 보인다.

표 1. sA2C 알고리즘 수도코드
Table 1. sA2C algorithm pseudocode

```
Input : one local network agent's parameters Θi
Output : global network's shared parameters Θglobal
1. Initialize local network : sA2C local network
2. Initialize environment, state, done
4. while global episode < max episode do
5.      action = get action(state)
6.      done, reward, next state = env.step(action)
7.      Update global values with lock
8.      train model(state, action, reward, next state,
done)
9.      if done then
10.         Increment global episode
11.     end if
12. end while
13. function train model(state, action, reward, next
state, done)
14.     Calculate π, value from local network
15.     target value = reward + γ · (1-done) · value
16.     criticr loss = 1/2(target valeu  -  value)2
17.     advantage = target value - value
18.     actor loss = -Eπ(logπ(a|s) · advantage)
19.     Calculate Q-value
20.     Zero global optimizer gradient
21.     Backward total loss
22.     Q values = global Q values[:]
23.     softmax weights = softmax(Q values)
24.     importance = softmax weights[local network
id]
25.     Θadjusted = Θi · importcane
26.     Update global network parameters using
Θadjusted
27.     Load Θglobal into local network
28. end function
```

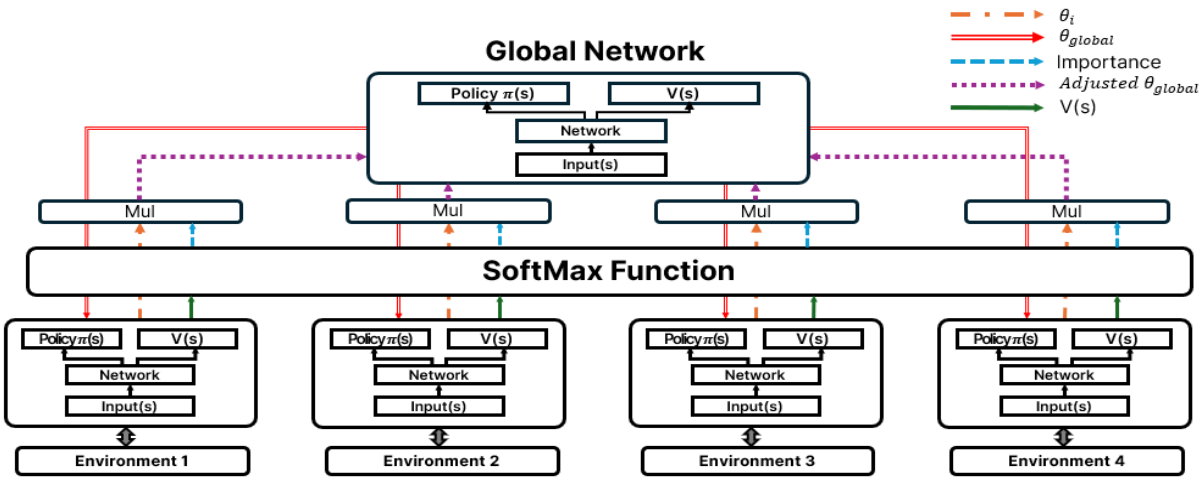

그림 2. sA2C 알고리즘 아키텍처
Fig. 2. sA2C algorithm architecture

이 내용을 자세히 서술하자면, 각 local network에서는 학습을 진행하면서 상태 가치함수인 V(s)를 구한다. V(s)는 각 local network에서 특정 상태일 때 에이전트가 얻을 수 있는 총 보상의 기댓값을 의미하며, 이 값은 학습이 좋은 방향으로 진행하고 있는지에 대한 척도로 사용된다. 각 local network에서 구한 V(s)값을 다른 환경의 local network와 비교하기 위해서 Softmax 함수를 사용하여 각 환경의 V(s) 값의 합이 1이 되게 V(s)를 변형시킨다. 변형된 V(s) 값은 모든 환경에서의 V(s)값을 비교하였기 때문에, 기존 V(s)값이 낮게 측정된 환경에서는 변형된 V(s) 값이 낮아서 각 환경에 대한 중요도로 나타낼 수 있다. 이러한 중요도들을 반영하여 각 local network에서는 학습한 파라미터 값에 중요도를 반영하여 global network에 전달하게 된다. 그러면 V(s)값이 낮은 환경에서는 중요도가 낮게 편성되어 모델의 학습 성능을 떨어지는 것을 방지할 수 있다. 이러한 학습 알고리즘을 통해 험지 환경에서 자율주행 학습을 진행할 때, 목표 지점으로 이동하지 않는 로봇은 중요도의 수치가 낮은 상태로 global network에 반영하게 되고 반대로 중요도가 높은 로봇은 global network에 높게 반영되어 자율주행 모델의 학습 성능을 높일 수 있다.

3.2 기본 환경 구축

이 논문에서는 험지 환경의 예시로 복분자 농가를 선택하였다. 복분자 나무는 너무 많은 물을 흡수하게 되면 쉽게 썩는다는 특징을 가지고 있다. 이 특징으로 인해, 복분자 환경에서는 경사가 있는 환경으로 구성하거나 땅을 다듬지 않는 형태인 오프로드 환경으로 구성하여 비가 와도 최대한 물을 빠지는 형태로 환경을 만드는 것이 중요하다. 이처럼 복분자 환경은 복잡하고 험난한 험지 환경의 조건에 만족하는 환경이라 할 수 있다. 이러한 복분자 환경에서 시뮬레이션으로 자율주행 학습을 진행하기 위해, unity 가상환경을 사용하여 그림 3같이 환경을 구성하였다.

기본 복분자 환경을 설명하자면, 가장 기본적인 에셋으로 자율주행 학습을 진행하는 agent와 특정 목표 지점 역할을 하는 target, 장애물 역할을 하는 복분자 에셋인 obstacle, 전체 환경인 environment가 존재한다. 구성된 복분자 환경에서 Episode가 시작하게 되면, agent 역할을 하는 로봇은 설정된 초기 위치에서 무작위로 위치된 target에 도달하기 위해 다양한 행동을 진행하게 된다. 이때 행동은 agent가 target에 최대한 빠르게 도달하기 위해서 본인의 행동 중 가장 최적의 행동을 찾는다. 이러한 학습을 진행하면서, 로봇은 목표 지점에 도달하기 위해서 계속 최적의 행동을 찾게 되고 만약 목표 지점에 도달하거나 장애물 혹은 환경에 벗어나게 되면 episode가 종료되고 환경이 재시작되면서 agent는 초기 위치, target은 무작위 위치로 이동하게 된다. 이를 반복함으로써 로봇은 episode마다 달라진 목표 지점까지 도달하기 위해, 장애물을 계속 피하고 최적의 경로로 이동하는 행동을 찾는 학습이 진행된다.

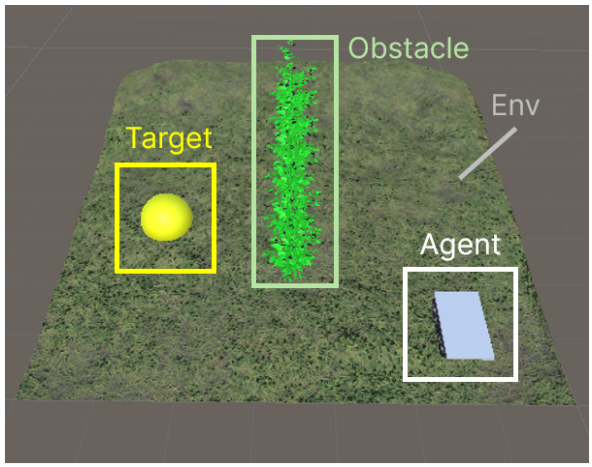

그림 3. 제안한 기본 unity 가상환경
Fig. 3. Proposed basic unity virtual environment

3.3 다중 복분자 환경 구축

실제 복분자 환경은 가상환경으로 만들어진 복분자 환경과 다르게 다양한 변인 요소를 가진다. 이 논문에서는 실제 환경에서 자율주행 학습에 가장 큰 영향을 미칠 수 있다고 생각된 변인 요소로 날씨, 수확물의 크기, 지형 형태를 총 3가지를 선택하였다. 표 2에서는 선택한 변인 요소를 현재 구성한 단일 가상환경과 실제 환경의 차이점을 나타낸다.

표 2. 단일 가상환경과 실제 환경의 차이점
Table 2. Differences between a single virtual environment and a real environment

Classification	Single virtual environment	Real environment
Weather	Clear weather	Clear weather, rainy weather
Crop size	Consistent size	Varying crop size each year
Terrain type	Uniform terrain	Terrain altered by specific factors

첫 번째로 날씨는 농가에서 가장 흔하게 발생하는 변인 요소이다. 복분자의 수확시기는 6월 말부터 7월 초로 매우 짧은 수확시기를 가진다. 짧은 수확시기로 인해 농부들은 맑은 날씨가 아니더라도 수확해야 해서 다양한 날씨에도 자율주행로봇이 사용해야 한다. 하지만 단일 가상환경에서는 일정한 날씨로 설정되어 있으므로 날씨에 대한 변인 요소를 학습하기가 어렵다. 이 점을 보완하기 위해, 비가 오는 환경에서도 학습하기 위한 가상환경을 추가하였다. 비가 오는 환경은 기존 환경에 비가 내리는 효과와 마찰력으로 인해 로봇의 이동속도를 감소하여 기존 학습과 다른 형태의 학습이 진행되도록 설계하였다. 이러한 특징으로 자율주행로봇은 같은 행동을 진행해도 기본 환경과 다른 위치로 이동되며, 이에 따라 행동에 대한 상태 값이 달라진다. 이 방식을 통해, 로봇은 비가 오는 환경에서 목표 지점에 도달할 수 있는 최적의 행동을 새롭게 찾아야 한다.

두 번째는 수확물의 크기로, 수확물은 농가에서 부딪히면 안 되는 장애물의 역할을 하고 있다. 하지만, 자율주행로봇은 주변에 수확물이 있는 고랑에서 이동해야 하며, 수확물에 피해를 주지 않기 위해 최대한 장애물을 피해야 한다. 그러나 수확물은 매해 성장에 따라 크기가 달라지며 고랑도 수확물에 따라 이동하는 폭이 달라진다. 이처럼 수확물이 다양한 크기로 존재하는 환경은 더 복잡한 경로 계획을 요구하며, 로봇의 충돌 회피 능력을 강화한다. 이를 적용하기 위해 수확물이 다른 환경을 구축하여 기본 환경보다 더 많은 수확물로 인해 더 커진 장애물의 형태를 가진 환경을 구성하였다. 커진 수확물은 agent의 이동할 수 있는 공간을 좁히며, agent는 좁아진 이동 범위 내에서 장애물을 회피하며 최적의 경로로 이동해야 한다.

마지막으로 지형 형태이다. 지형 형태는 다양한 특정 요인으로 지형이 달라지는 경우를 말한다. 예를 들어, 경운작업을 진행하면서 기존 지형 형태가 변하게 되며, 이는 기존에 학습한 지형과 달라진 지형을 가지게 된다. 이를 반영하기 위해, 기존 지형의 전체적인 크기나 높이는 동일하지만, 지형의 세부적인 경사나 형태를 변경한 새로운 가상환경을 구축하였다. 이로 인해, 지형이 달라진 환경은 기존 환경의 학습한 데이터와 전혀 다른 결과를 가져오게 되며, 목표 지점으로 가기 위해 기존 학습한 내용과 다른 행동이 최적의 행동으로 바뀔 수 있다.

결과적으로, 로봇은 다양한 지형에서의 적응력을 높여야 하며, 이는 실제 환경에서의 성능 향상으로 이어질 것이다. 이를 적용하기 위해서는 로봇은 달라진 지형에서도 학습을 진행하여 최적의 행동을 찾아야 한다.

그림 4. 변인 요소가 포함된 다중환경 구축
Fig. 4. Construction of multiple environments containing variable elements

이처럼 3가지의 변인 요소를 가진 다중 가상환경과 기본 환경을 포함한 4개의 가상 복분자 환경을 그림 4처럼 구성하였다. 이들은 서로 다른 변인 요소가 포함된 환경에서 특정 목표 지점까지 이동하는 최적의 행동을 찾는 학습이 진행되며, sA2C 알고리즘을 사용하여 복분자 환경에서 얻은 데이터를 최종 모델에 적용하는 시스템을 구축하였다.

Ⅳ. 실험 및 평가

4.1 학습 환경 설계 및 보상 함수 설계

복분자 환경을 학습하기 위해 agent의 상태 공간과 행동 공간을 설계하였다. 상태 공간의 데이터로는 로봇의 위치와 목표 지점의 위치, 로봇과 목표 지점의 거리를 저장하게 설계하였다. 이를 이용하여 로봇은 자신과 목표 지점의 위치, 거리를 통해 자신이 목표 지점에 얼마나 가까웠는지 판단할 수 있다. 행동 공간은 로봇이 움직일 수 있는 상하좌우로 설정하였다. 로봇은 위치를 통해, 목표 지점과 가까워지는 최적의 행동을 찾아 이동하게 된다. 보상 함수는 agent가 목표 지점에 가까워지는 것을 주된 목표로 설정하였다. 거리가 이전 거리보다 더 가까워질 경우, 가까워진 거리 * 0.1의 양의 보상 값을 부여하게 된다. 하지만 더 멀어졌을 경우, 멀어진 거리 * 0.05의 음의 보상 값으로 부여하여 보상 함수를 설계하였다. Episode가 종료되는 경우는 목표 지점에 도달했을 경우 +10의 양의 보상을 부여하는 경우와 벽이나 장애물에 부딪히는 경우 - 5의 음의 보상을 부여하는 경우로 설정하였다.

4.2 sA2C와 A3C 자율주행 실험 및 평가

sA2C와 A3C 알고리즘의 성능을 비교하기 위해 표 3에서는 학습에 사용한 하이퍼파라미터를 동일한 값으로 진행하였으며, 평가 지표로는 알고리즘의 actor loss, criticr loss, 평균 보상을 사용하였다.

그림 5는 각 알고리즘의 actor loss값을 의미한다. A3C 알고리즘의 초기 actor loss값은 0.0598을 나타내고 있으며 episode가 진행될수록 최대 0.0087까지 내려갔다. 그에 비해 sA2C 알고리즘은 초기에는 0.0969로 A3C보다 높은 actor loss값을 보여주지만, 100번째 episode가 진행되었을 때 sA2C 알고리즘이 더 낮은 actor loss값에 도달하였고. 최종적으로는 actor loss가 0.0045까지 감소하여 기존 A3C보다 48.28% 낮은 것을 확인할 수 있었다.

표 3. 학습에 사용된 파라미터
Table 3. Parameters used for training

Parameter	Value
Discount factor	0.90
Learning rate	0.000025
Maximum steps per episode	50000
Maximum episode	500
Hidden layer1	128
Hidden layer2	64
Actor size	4
Episode	500

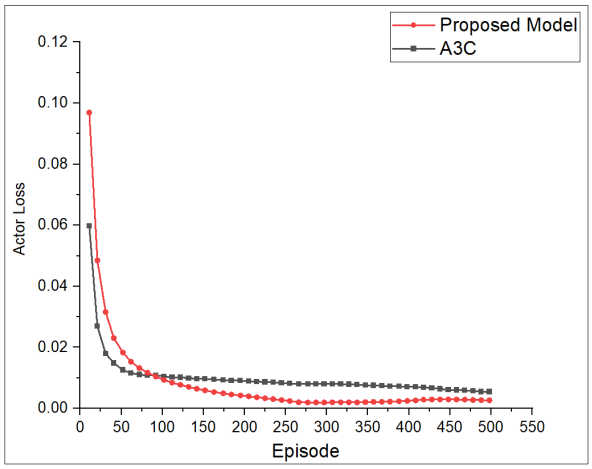

그림 5. sA2C와 A3C의 actor loss 그래프
Fig. 5. Actor loss graph of sA2C and A3C

그림 6은 criticr loss의 값을 비교하였다. A3C 알고리즘은 초깃값은 0.9140의 값을 가지며 50 episode가 진행하면서 0.5에 loss 값이 수렴하였으며, 최종적으로 최소 0.4811까지 감소하였다. 하지만 sA2C 알고리즘은 초깃값이 2.8821로 매우 높은 criticr loss 값을 보였지만 100번째 episode에 A3C 보다 criticr loss가 더 낮게 수렴하였으며, 지속적인 감소로 최소 0.2663까지 감소하여 A3C 알고리즘보다 44.63% 낮게 수렴한 것을 확인할 수 있었다.

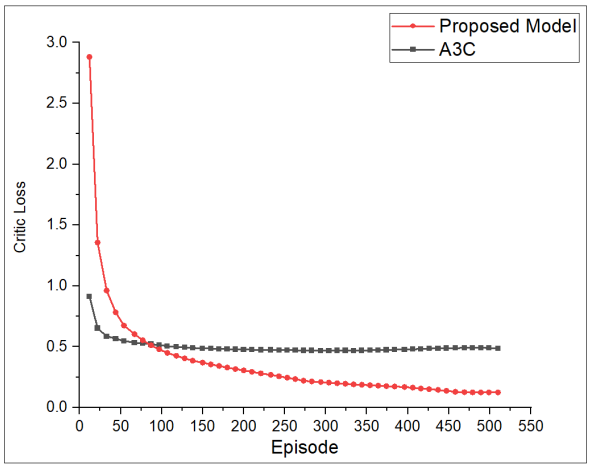

그림 6. sA2C와 A3C의 criticr loss 그래프
Fig. 6. Criticr loss graph of sA2C and A3C

마지막으로 그림 7은 episode에서의 step당 평균 보상 값을 측정했다. A3C 알고리즘은 -0.0413으로 보상이 시작되어 episode가 진행될수록 보상 값이 증가하였지만, loss값이 100번째 이후로 loss 값이 수렴함으로써 보상 값이 최대 0.1757까지 증가하였다. 하지만 sA2C 알고리즘은 -0.1449으로 시작하였으며, 지속해서 loss값이 감소하여 보상 값이 최대 0.2513까지 증가하였다. 최종적으로 지속적인 Loss 값의 감소로 인해 sA2C 알고리즘의 평균 보상이 점차 증가하였으며 400번째 episode에서는 A3C 알고리즘보다 더 높은 보상을 획득하였고 최대 보상 값은 43.09% 차이를 보이며 더 높은 보상 값을 가졌다.

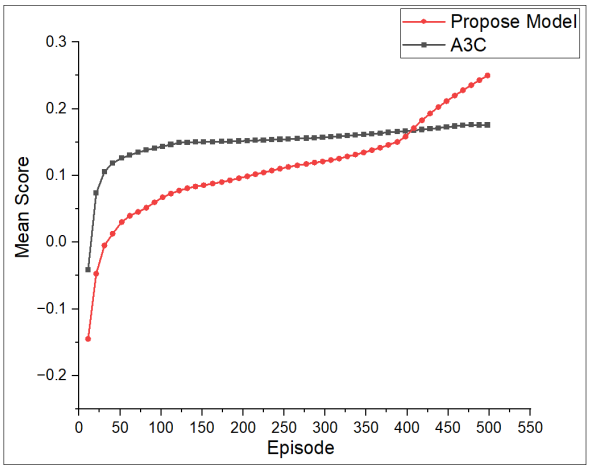

그림 7. sA2C와 A3C의 평균 보상 그래프
Fig. 7. Average reward graph of sA2C and A3C

실험 결과로 A3C 알고리즘은 100번째 episode 이후로 actor loss값과 criticr loss값이 빠르게 수렴하는 값에 도달하여 평균 보상 값이 낮은 값에 수렴하게 되었다. 하지만 sA2C 알고리즘은 A3C 알고리즘에 비해 100번째 episode 이후에도 지속해서 loss값이 감소하였으며, 최종적으로 400번째 episode에서는 평균 보상 값이 더 높은 값을 가지게 되었다. 이를 통해 sA2C 알고리즘이 A3C 알고리즘보다 더 좋은 성능을 보여주고 보상 측면에서도 더 높은 값을 가지게 되는 것을 확인할 수 있었다.

표 4. 각 알고리즘의 손실 값과 평균 보상 값
Table 4. Loss and average compensation values for each algorithm

Classification	A3C algorithm	Proposed model
Maximum actor loss	0.0598	0.0969
Minimum actor loss	0.0087	0.0045
Maximum criticr loss	0.9140	2.8821
Minimum criticr loss	0.4811	0.2663
Maximum mean score	0.1757	0.2513

V. 결론 및 향후 과제

이 논문에서는 기존 A3C 알고리즘의 문제점을 보완한 sA2C 알고리즘 강화학습 방법을 제안하고 다중 험지 환경을 구축하여 다양한 변인 요소를 학습하는 시스템을 제안하였다. 다중환경에서는 기본적인 복분자 환경과 비가 오는 환경, 수확물의 크기가 커진 환경, 지형이 달라진 환경으로 총 4가지의 환경을 구성하였다. sA2C 알고리즘으로 구현한 자율주행 모델 학습 실험을 진행하였으며, 실험 결과를 global network의 actor loss와 criticr loss, step당 평균 보상 값을 측정하고 비교하였다. 이를 A3C 알고리즘으로 구현한 방법과 제안한 sA2C 알고리즘에 대한 성능 비교 테스트를 진행하였다. 실험 결과 다중 험지 환경에서 sA2C 알고리즘의 actor loss값과 criticr loss값이 46% 더 낮게 수렴하였고 평균 보상은 sA2C 알고리즘이 43%로 더 높게 더 높게 측정되었다.

이처럼 다중 험지 환경에서 sA2C 알고리즘이 A3C 알고리즘보다 더 높은 학습 성능을 보여줬다.

이 논문에서는 각 환경에서 학습 성능을 비교하고 중요도를 구하여, 학습 파라미터를 중요도만큼 공유 모델에 반영하여 모델의 학습 효율을 높이는 sA2C 알고리즘을 제안하였다. 추가로 sA2C 알고리즘과 기존 A3C 알고리즘을 다중 험지 환경에서 성능 비교실험을 진행하였다. 추후에는 sA2C 알고리즘을 활용하여 가상환경에 동적인 장애물을 추가함으로써 실제 환경과 더 유사한 시뮬레이션 환경을 구축할 예정이며, DDPG 알고리즘이나 dreamer 알고리즘과 같은 다양한 멀티에이전트 강화학습 방법이나 추가적인 vision 알고리즘과 같이 적용하여 험지 환경에서의 자율주행에 적합한 시뮬레이션 학습 방법을 검토할 예정이다.

References

[1] C. E. Shannon, "A mathematical theory of communication", The Bell System Technical Journal, Vol. 27, No. 3, pp. 379-423, Jul. 1948. https://doi.org/10.1002/j.1538-7305.1948.tb01338.x.

[2] S. Li, L. D. Xu, and S. Zhano, "The internet of things: a survey", Information Systems Frontiers, Vol. 17, pp. 243-259, Apr. 2014. https://doi.org/10.1007/s10796-014-9492-7.

[3] V. Mnih, et al., "Playing atari with deep reinforcement learning", arXiv preprint arXiv:1312.5602, Dec. 2013. https://doi.org/10.48550/arXiv.1312.5602.

[4] M. Bojarski, et al., "End to end learning for self-driving cars", arXiv preprint arXiv:1604.07316, Apr. 2016. https://doi.org/10.48550/arXiv.1604.07316.

[5] D. Lee, "A Comparative Analysis of Reinforcement Learning Activation Functions for Parking of Autonomous Vehicles", The Journal of The Institute of Internet, Broadcasting and Communication, Vol 22., No. 6, pp. 75-81, Dec. 2022. https://doi.org/10.7236/jiibc.2022.22.6.75.

[6] H. Hu, K. Zhang, A. H. Tan, M. Ruan, C. Agia, and G. Nejat, "A sim-to-real pipeline for deep reinforcement learning for autonomous robot navigation in cluttered rough terrain", IEEE Robotics and Automation Letters, Vol. 6, No. 4, pp. 6569-6576, Oct. 2021. https://doi.org/10.1109/LRA.2021.3093551.

[7] S. Jo, R. Kwon, and G. Kwon, "Safety Evaluation for Reinforcement Learning Model: Case Study of Autonomous Driving", The Journal of Korean Institute of Information Technology, Vol. 21, No. 8, pp. 165-174, Aug. 2023. https://dx.doi.org/10.14801/jkiit.2023.21.8.165.

[8] S. Hossain and D. J. Lee, "Autonomous-driving vehicle learning environments using unity real-time engine and end-to-end CNN approach", Journal of Korea Robotics Society , Vol. 14, No. 2, pp. 122-130, Jun. 2019. https://doi.org/10.7746/jkros.2019.14.2.122.

[9] V. Mnih, et al., "Asynchronous methods for deep reinforcement learning", arXiv preprint arXiv:1602.01783, Vol. 48, pp. 1928-1937, Feb. 2016. https://doi.org/10.48550/arXiv.1602.01783.

[10] A Sharif and D. Marijan, "Evaluating the robustness of deep reinforcement learning for autonomous policies in a multi-agent urban driving environment", 2022 IEEE 22nd International Conference on Software Quality, Reliability and Security (QRS), Guangzhou, China, Vol. 20, pp. 785-796, Dec. 2022. https://doi.org/10.1109/QRS57517.2022.00084.

[11] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms", arXiv preprint arXiv:1707.06347, Jul. 2017. https://doi.org/10.48550/arXiv.1707.06347.

[12] L. Espeholt et al., "Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures", In International conference on machine learning, Vol. 80, pp. 1407-1416, Jul. 2018.

저자소개

장 승 균 (Seungkyun Jang)

2019년 3월 ~ 현재 :
국립군산대학교 소프트웨어학과
학사과정
관심분야 : 게임, 강화학습

정 동 원 (Dongwon Jeong)

1997년 2월 : 군산대학교
컴퓨터과학과(학사)
1999년 2월 : 충북대학교
전자계산학과(석사)
2004년 2월 : 고려대학교
컴퓨터학과(박사)
2005년 4월 ~ 현재 :
국립군산대학교 소프트웨어학과 교수
관심분야 : 데이터베이스, 시맨틱 서비스, 빅데이터,
사물인터넷, 엣지컴퓨팅, 지능형 융합 서비스

온 병 원 (Byung-Won On)

2007년 8월 :
펜실베이니아주립대학교
컴퓨터공학과(박사)
2008년 2월 :
브리티시컬럼비아대학교
컴퓨터과학과(박사 후 연구원)
2010년 9월 : 일리노이대학교
ADSC센터 선임연구원
2011년 9월 : 서울대학교 차세대융합기술연구원 연구교수
2014년 4월 ~ 현재 : 국립군산대학교 소프트웨어학과
교수
관심분야 : 데이터 마이닝, 인공지능, 강화학습

정 현 준 (Hyunjun Jung)

2008년 3월 : 삼육대학교
컴퓨터과학과(공학사)
2010년 3월 : 숭실대학교
컴퓨터학과(공학석사)
2017년 9월 : 고려대학교
컴퓨터·전파통신공학과(공학박사)
2017년 8월 ~ 2020년 8월 :
광주과학기술원 블록체인인터넷경제연구센터 연구원
2021년 3월 ~ 현재 : 국립군산대학교 소프트웨어학과
교수
관심분야 : 블록체인, 데이터 사이언스, 센서 네트워크,
사물인터넷, 머신러닝