Journal of KIIT. Vol. 24, No. 6, pp. 157-167, Jun. 30, 2026. pISSN 1598-8619, eISSN 2093-7571
http://dx.doi.org/10.14801/jkiit.2026.24.6.157

다중 환경에서의 자율 주행을 위한 궤적 중요도 가중 기반 LLM 보강 A2C

김동연*, 정현준**

LLM-Enhanced A2C with Trajectory Importance Weighting for Diverse Environments Autonomous Driving

Dongyeon Kim*, Hyunjun Jung**

요 약

기존 A2C 자율주행 시스템은 희소한 보상 구조로 인한 느린 수렴과 다양한 조건에 대한 성능 문제가 있다. 이를 개선하기 위해 환경별 경험을 구분하여 학습 효율을 높이는 sA2C-T 방법과 LLM 힌트를 활용한 상황 기반 강화학습 방법이 제안되었으나, 고정된 임계값 기반의 필터링으로 인해 다중환경에서 성능차이가 있다. 이 연구는 대형 언어 모델(LLM)의 추론 능력과 A2C의 실시간 학습 능력을 통합한 궤적 중요도 가중 기반 LLM 보강 A2C 학습 방법을 제안한다. LLM은 행동 조언, 적응적 난이도 조절, 궤적 중요도 평가를 통해 학습을 최적화한다. 실험 결과, 모든 날씨 조건에서 기존 대비 평균 15.5%의 보상 향상과 3.1%의 성공률 상승을 확인하였으며, 이는 LLM이 A2C의 학습에 도움이 되었음을 보여준다.

Abstract

Existing A2C autonomous driving systems suffer from slow convergence due to sparse reward structures and performance issues under diverse conditions. To address these, sA2C-T methods for distinguishing environment-specific experiences and context-based RL using LLM hints have been proposed, but they exhibit performance variance across multiple environments due to fixed threshold-based filtering. This study proposes a trajectory importance-weighted LLM-augmented A2C learning method integrating LLM reasoning and A2C real-time learning. The LLM optimizes learning through action advice, adaptive difficulty adjustment, and trajectory importance evaluation. Experimental results confirmed an average reward improvement of 15.5% and a 3.1% success rate increase across all weather conditions, demonstrating that LLM contributes to enhancing A2C learning performance.

Keywords
AI, LLM, reinforcement learning, A2C

* 국립군산대학교 소프트웨어학과 학사과정
- ORCID: https://orcid.org/0009-0006-8048-2696
** 국립군산대학교 소프트웨어학과 교수(교신저자)
- ORCID: https://orcid.org/0000-0002-6717-1395

· Received: Mar. 10, 2026, Revised: May 05, 2026, Accepted: May 08, 2026
· Corresponding Author: Hyunjun Jung
Dept. of 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 AI 기술의 발전으로 인해 다양한 분야에서 자율주행 시스템이 제안되었다[1]. 자율주행 시스템은 복잡하고 동적인 환경에서 안전하고 효율적인 실시간 의사결정을 요구하는 안전 시스템이다. 강화학습은 환경과의 상호작용을 통해 최적 정책을 학습하는 방법론으로, 전통적인 규칙 기반 시스템의 한계를 극복하는 방법으로 제시되었다[2]. 강화학습이란 학습의 주체인 에이전트가 특정 목표를 달성하기 위해 주어진 환경에서 상태에 따른 행동을 수행하고, 환경으로부터 받은 보상을 통해 최적의 정책을 학습하는 방법론이다. 이러한 학습 방식을 통해 에이전트는 다양한 상황에서 목표에 가장 적합한 최적의 행동을 학습하게 된다. 강화학습 알고리즘 중 하나인 A2C(Advantage Actor-Critic)는 정책 함수와 가치 함수를 동시에 학습함으로써 안정적이고 효율적인 학습을 가능하게 하는 정책 기반 알고리즘이다[3]. 자율주행 분야에서도 A2C를 적용한 연구들이 진행되었고, 종방향 제어 작업에서 높은 성능을 보이는 것으로 보고되었다[4]. 하지만 기존의 A2C 기반 자율주행 시스템에서는 여러 가지 문제점이 발견되었다. 첫째, 복잡한 교통 환경에서의 희소한 보상 구조로 인해 학습 수렴 속도가 느리고 불안정한 경우가 빈번하다[5]. 희소 보상 환경에서는 무작위 행동을 통한 탐험으로 목표를 달성하는 횟수가 현저히 낮아 희소 보상 시스템을 만날 확률이 높아진다. 특히 자율주행과 같은 경로 계획 문제에서는 로봇 내비게이션을 위한 탐험에서 목표 달성 횟수가 현저히 낮기 때문에 이러한 문제가 두드러진다. 둘째, 다양한 날씨 조건과 환경 변화에 대한 일관성 있는 성능이 부족하다[6]. 기존 자율주행 시스템들은 특정 환경 조건에서만 학습되어 실제 운행 환경에서 마주칠 수 있는 다양한 기상 조건에서의 성능 저하가 관찰되었다. 특히 다중 환경에서의 학습 성능을 높이기 위해 임계값 필터링을 활용한 sA2C-T 방법[7]이 제안되었으나, 고정된 임계값 사용으로 인한 적응성 부족이라는 한계가 존재한다. 각 환경의 동적인 특성과 학습 진행 상황을 반영하지 못해, 환경별 최적의 학습 효율을 달성하기 어렵다는 문제가 있다. 셋째, 안전 중요 상황에서의 위험 인식과 대응 능력이 제한적이다[8][9]. 강화학습 기반 자율주행 시스템에서는 탐험과 활용의 균형을 맞춰야 하는데, 안전이 중요한 환경에서 탐험 중 발생할 수 있는 위험한 행동이 문제가 될 수 있다. 대형 언어 모델(LLM, Large Language Model)의 발전은 자율주행 분야에 새로운 가능성을 제시하였다[10]. LLM은 방대한 텍스트 데이터를 통해 학습된 일반화된 지식과 추론 능력을 보유하고 있으며, 복잡한 상황에 대한 설명 가능한 의사결정을 제공할 수 있다. 궤적 중요도 평가는 강화학습에서 학습 데이터의 품질을 향상시키기 위하여 수집된 경험 데이터 중에서 학습에 더 유용한 궤적을 식별하고 가중치를 부여하는 방법이다[11]. 특히 자율주행과 같은 복잡한 도메인에서는 안전 중요 상황이나 복잡한 의사결정이 요구되는 궤적이 단순한 직진 주행 궤적보다 더 중요한 학습 가치를 가질 수 있다[12]. 이러한 문제점들을 해결하기 위해 이 연구에서는 LLM의 추론 능력을 활용한 동적 궤적 중요도 평가를 통해 고정 임계값의 한계를 극복하고, A2C의 실시간 학습 능력을 통합한 "LLM기반 궤적 중요도를 활용한 A2C"라는 학습 방법을 제안한다. 제안하는 방법은 LLM 기반 행동 조언기, 적응적 난이도 조절기, 궤적 평가기를 포함하는 통합 프레임워크로 구성되며, n개의 서로 다른 환경에서 병렬 학습을 통해 일관된 성능을 확보한다. LLM 기반 행동 조언기는 현재 상황을 자연어로 분석하여 최적의 행동을 제안함으로써 에이전트의 탐험 효율성을 개선하고, 적응적 난이도 조절기는 커리큘럼 러닝을 기반으로 에이전트의 최근 성과를 실시간으로 분석하여 환경의 난이도를 동적으로 조절한다. 궤적 평가기는 완료된 에피소드의 궤적을 분석하여 학습 가치를 평가하고 경험 재생 버퍼의 중요도 가중치를 동적으로 조정함으로써, 기존 고정 임계값 방식의 적응성 문제를 해결한다. 제안하는 시스템의 성능을 확인하기 위해 Clear Weather, Light Rain, Heavy Rain, Fog의 4가지 날씨 조건으로 구성된 다중 날씨 환경을 구축하여 기존 A2C 알고리즘과 제안한 LLM-Enhanced A2C 알고리즘의 성능 비교 실험을 진행하였다. 각 환경별로 시야 제한과 차량 제어 난

이도가 점진적으로 증가하도록 설계하였으며, LLM Response Cache를 통해 동일한 상황에 대한 반복적인 API 호출을 방지하여 비용 효율성을 높였다.

이 논문의 2장에서는 A2C 알고리즘과 LLM을 활용한 자율주행 관련 연구를 소개하여 제안 방법의 이론적 배경에 관해 설명한다. 3장에서는 LLM-Enhanced A2C 시스템에 대한 상세한 설명과 다중 날씨 환경 구축 방법을 소개한다. 4장에서는 구축된 다중 날씨 환경을 통해 기존 A2C 알고리즘과 제안한 LLM-Enhanced A2C 알고리즘의 성능 비교를 진행하여 각 알고리즘의 손실 값과 평균 보상 값을 측정하였다. 5장에서는 최종적으로 전체 논문에 대한 요약과 결론을 논하며 향후 과제에 관해 설명한다.

이 논문의 주요 기여는 세 가지이다. 첫째, 기존 sA2C-T의 고정 임계값 방식과 달리 LLM의 상황적 추론 능력을 활용하여 경험 데이터의 중요도를 동적으로 산출하는 궤적 중요도 가중 메커니즘을 제안한다. 둘째, LLM 행동 조언기, 적응적 난이도 조절기, 궤적 평가기를 단일 프레임워크로 통합하여 학습 전 과정에서 LLM이 보조적 역할을 수행하는 구조를 구현한다. 셋째, 기존 HCRMP[13]가 상태 표현 개선에 집중한 것과 달리 이 연구는 경험 재생 버퍼의 우선순위 가중 조정을 통해 학습 데이터 품질을 향상시키는 새로운 접근을 제시한다.

II. 관련 연구

2.1 임곗값 필터링을 활용한 sA2C-T 강화학습 방법

이 논문은 다중 에이전트 강화학습에서 각 에이전트의 학습 성능에 따른 중요성 가중치를 공유 모델에 반영할 때, 성능이 지나치게 떨어지는 학습 환경도 공유 모델에 반영되어 전체 학습 안정성을 저하시키는 문제를 지적한다[7]. 이를 해결하기 위해 중요성 가중치에 임곗값 필터링 방법을 적용한 sA2C-T 알고리즘을 제안하였다. 각 환경에서 구한 가치 함수 V(s) 값을 정규화하여 중요성 가중치로 변환하고, 사용자가 설정한 임곗값 이상인 환경의 데이터만 Global Network에 반영하도록 설계하였다. Unity 가상환경에서 다중 험지 환경 자율주행 실험을 통해 기존 sA2C 알고리즘 대비 8.57% 향상된 보상 값 수렴을 달성했다. 다만 고정된 임곗값 사용으로 인한 적응성이 부족하다는 한계가 있다, 이 연구에서는 LLM의 상황적 추론 능력을 활용하여 동적인 궤적 중요도 평가를 제공하고자 한다.

2.2 LLM 기반 정책 탐색을 활용한 추천 시스템 강화학습

이 논문은 추천 시스템에서 오프라인 강화학습 정책이 정적 사용자 데이터로 훈련되어 동적 온라인 환경에 배포될 때 분포 이동에 취약하고, 단기적으로 관련성이 높은 아이템에만 집중하여 탐색을 저해하는 문제를 지적한다[14]. 이를 해결하기 위해 LLM을 활용하여 사용자 목표와 선호도를 모방하고 정책을 오프라인에서 사전 훈련하는 iALP(Interaction-Augmented Learned Policy) 프레임워크를 제안하였다. LLM에 사용자 상태를 프롬프트로 제공하여 아이템 선호도를 추출하고, 피드백을 기반으로 보상을 학습하며, Actor-Critic 프레임워크를 사용하여 정책을 업데이트한다. 온라인 배포를 위해 적응형 변형인 A-iALP를 도입하여 정책 타협과 제한된 탐색 문제를 완화하였다. 다만 자율주행과 같은 연속 제어 환경에서의 적용성이 검증되지 않았으며, 궤적 수준의 중요도 평가가 부족하다. 이 연구에서는 이러한 한계를 보완하여 자율주행 도메인에서 LLM 기반 궤적 중요도 평가를 통한 정책 학습을 구현하고자 한다.

2.3 LLM 힌트를 활용한 상황 기반 강화학습

이 논문은 기존 LLM 기반 강화학습 기법들이 언어 모델의 환각(hallucination)에 지나치게 의존하여 안정적인 주행 정책 학습에 한계를 보인다는 점을 지적한다[13]. 이를 해결하기 위해 HCRMP(LLM-Hinted Contextual Reinforcement Learning Motion Planner) 프레임워크를 제안하였으며, LLM이 생성한 의미적 힌트를 상태 표현에 보강하고 지식 기반 안정화 모듈을 추가하여 불확실한 정보를 억제하였다. 또한,

LLM 가이드라인은 저빈도 업데이트로, RL 제어는 고빈도 실행으로 분리하여 실시간 제어를 가능하게 하였다. CARLA 시뮬레이터에서의 실험 결과, 제안 기법은 최대 80.3%의 성공률과 11.4%의 충돌률 감소를 기록하며 안전성과 강건성을 향상시켰다. 다만 이 연구는 정책 안정성과 상태 표현 개선에 중점을 두고 있어 학습 과정에서의 궤적 품질 평가나 동적 중요도 가중치 조정에는 상대적으로 제한적이며, 학습 데이터의 세밀한 중요도 구분에는 한계가 있다.

III. LLM기반 궤적 중요도를 활용한 A2C

3.1 전체 아키텍처

그림 1은 제안하는 시스템의 전체 아키텍처를 보여준다. 제안하는 시스템은 LLM Intelligence Layer, Agent & Memory Layer, Environment Layer로 구성된다. LLM Intelligence Layer는 LLM Action Advisor, LLM Difficulty Controller, LLM Trajectory Evaluator로 구성되며, LLM Response Cache를 통해 동일한 상황에 대한 중복 호출을 방지한다. Agent & Memory Layer는 A2C Agent와 경험 재생 버퍼를 포함하며, Environment Layer는 Environment Manager가 Clear, Rain, Fog 등 다양한 날씨 조건의 병렬 환경들을 관리한다. A2C는 본래 on-policy 알고리즘으로 경험 재생 버퍼를 직접 사용하지 않는다. 그러나 이 연구에서는 에피소드 단위로 완성된 궤적을 버퍼에 저장한 뒤, LLM이 산출한 중요도 가중치를 우선순위로 활용하여 학습에 선택적으로 반영하는 방식을 사용하였다. 이는 순수한 off-policy 재생이 아니라 에피소드 완료 후 단기 배치를 구성하는 방식으로, 정책 파라미터가 크게 변화하기 전의 경험만을 대상으로 하여 on-policy 특성을 근사적으로 유지한다. 이러한 설계는 희소 보상 환경에서 중요 경험을 반복 학습하기 위한 실용적 타협으로, 유사한 접근이 다중 환경 강화학습 연구[6][7]에서도 활용된 바 있다. 시스템의 전체적인 흐름은 다음과 같다. A2C 에이전트가 Environment Manager에서 관리하는 각기 다른 환경으로부터 관찰값을 수집하고, LLM Action Advisor로부터 행동 조언을 받아 환경에서 실행한다. 에피소드가 완료되면 LLM Trajectory Evaluator가 전체 궤적을 분석하여 학습 가치를 평가하고, 주기적으로 LLM Difficulty Controller가 에이전트의 성과를 분석하여 환경의 난이도를 동적으로 조절한다.

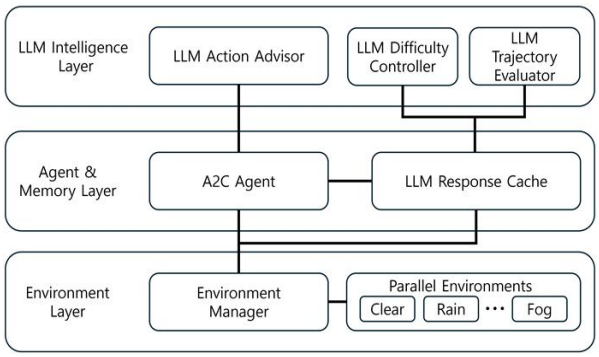

그림 1. LLM-Enhanced A2C 시스템 전체 아키텍처
Fig. 1. Overall architecture of LLM-Enhanced A2C system

3.2 LLM 기반 행동 조언 프로세스

LLM Action Advisor는 에이전트의 현재 관찰값을 분석하여 최적의 행동을 제안한다.

그림 2는 LLM 행동 조언 프로세스의 흐름을 나타낸다. 에이전트가 환경으로부터 관찰값을 얻으면, 먼저 LLM Response Cache에서 이전에 유사한 상황에서 제공된 조언이 있는지 확인한다. 캐시에 기존 응답이 있으면 해당 조언을 반환하여 LLM 호출을 절감한다. 캐시에 없는 경우(Miss)에만 새로운 LLM 호출을 통해 조언을 요청한다. LLM은 차량의 위치, 회전각, 목표 지점 정보를 분석하여 행동을 추천한다. 관찰값 정규화 과정에서 위치 정보는 소수점 2자리, 각도 정보는 소수점 1자리로 반올림하며, 정규화된 관찰값은 해시 기반 캐시 키를 생성하여 중복 호출을 방지한다. 에이전트는 LLM의 조언을 참고하여 행동을 선택하고, 선택된 행동을 환경에서 실행한다. 모든 과정은 기록되어 관찰값, 행동, 보상 정보가 궤적 데이터로 저장된다.

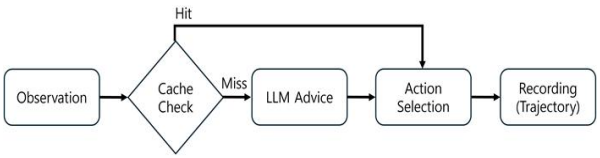

그림 2. LLM 행동 조언 프로세스
Fig. 2. LLM action advising process

LLM의 행동 조언은 소프트 가이던스(Soft guidance) 방식으로 정책 결정에 반영된다. LLM은 현재 상태에서 권장 행동 방향을 0(좌회전), 1(직진), 2(우회전), 3(후진) 중 하나로 출력하며, A2C 에이전트의 행동 확률 분포에서 조언 일치 행동의 로짓(logit)에 작은 보너스 값을 더하는 방식으로 탐험 방향을 유도한다. 이는 A2C의 정책 네트워크 파라미터를 직접 수정하는 것이 아니라 단계별 행동 선택 시 확률 편향을 부여하는 방식이므로, LLM 출력의 오류가 발생하더라도 정책 자체가 손상되지 않으며 탐험의 다양성도 유지된다. 알고리즘 1은 LLM 기반 행동 조언기의 전체 동작을 수도코드로 나타낸다.

알고리즘 1. LLM 행동 조언 수도코드
Algorithm 1. LLM action advising pseudocode

```
Input : vehicle_pos (x, y), heading_angle (deg),
        target_pos (x, y), dist_to_target,
        obstacles [ {x1, y1}, {x2, y2}, ... ],
        llm_client, cache
Output: action_advice

1. normalized_obs ← normalize_observation(vehicle_pos, heading_angle,

target_pos, dist_to_target)
2. obs_list ← sort_by_distance(obstacles, vehicle_pos)
3. nearest_dist ← euclidean(vehicle_pos, obs_list[0])
4. normalized_obs ← append(normalized_obs, obs_list, nearest_dist)
5. cache_key ← generate_hash(normalized_obs)
6. if cache.has(cache_key):
7.     return cache.get(cache_key)
8. prompt ← generate_situation_prompt(normalized_obs)
        // Prompt Template:
        // "You are an autonomous driving assistant.
        //  Analyze the current driving situation and recommend the best action.
        //
        //  Current State:
        //    - Vehicle position  : ({x:.2f}, {y:.2f})
        //    - Heading angle     : {angle:.1f} degrees
        //    - Target position   : ({tx:.2f}, {ty:.2f})
        //    - Distance to target: {dist:.2f}
        //
        //  Nearby Obstacles (sorted by distance):
        //    - Obstacle 1: position ({o1.x:.2f}, {o1.y:.2f}), distance {d1:.2f}  <- WARNING: very close
        //    - Obstacle 2: position ({o2.x:.2f}, {o2.y:.2f}), distance {d2:.2f}
        //    - ...
        //
        //  Available Actions:
        //    0 = turn left  |  1 = go straight  |  2 = turn right | 3 = reverse
        //
        //  Respond with only a single integer (0, 1, 2 or 3)."
9. advice ← llm_client.call(prompt)
        // Output Example: 2
        //  (우회전: 좌측 근접 장애물(d=1.80) 회피 후 목표 방향으로 진행)
10. cache.store(cache_key, advice)
11. return advice
```

3.3 적응적 난이도 조절 프로세스

LLM Difficulty Controller는 에이전트의 학습 성과를 주기적으로 분석하여 환경의 난이도를 동적으로 조절한다. 그림 3은 적응적 난이도 조절 루프의 구조를 나타낸다. n번의 에피소드마다 실행되며, 최근 에피소드의 성과 이력을 저장하고 성공률, 평균 보상, 에피소드 길이 등의 메트릭을 수집하여 평가한다. LLM은 수집된 메트릭들을 자연어 프롬프트로 받아 학습 상황을 분석하고, 난이도 상승, 유지, 하강 중 하나를 선택하여 응답한다. Environment Manager가 이 결정에 따라 환경 난이도를 조정한다. 알고리즘 2는 난이도 조절기의 수도코드이다.

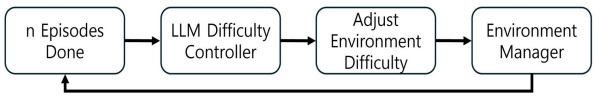

그림 3. 난이도 조절 루프
Fig. 3. Difficulty control loop

알고리즘 2. 난이도 조절 수도코드
Algorithm 2. Difficulty controlling pseudocode

```
Input : performance_metrics { success_rate, avg_reward, avg_ep_length },
  recent_history [ episode_1, ..., episode_n ],
        current_difficulty, llm_client, cache
Output: difficulty_decision ∈ { increase, maintain, decrease }
```

```
1. cache_key ← generate_cache_key(performance_metrics, recent_history)
2. if cache.has(cache_key):
3.     return cache.get(cache_key)
4. prompt ← format_performance_prompt(performance_metrics,

recent_history, current_difficulty)
        // Prompt Template:
        // "You are a curriculum controller for autonomous driving RL.
        //
        // Recent performance over last {n} episodes:
        //   - Success rate : {success_rate}%
        //   - Average reward : {avg_reward:.1f}
        //   - Avg episode length : {avg_ep_length} steps
        //   - Current difficulty : {current_difficulty}
        //
        // Based on this performance, decide the difficulty adjustment.
        // Respond with only one word: increase, maintain, or decrease."
5. response ← llm_client.call(prompt)
        // Output Example: "increase"
        //   (성공률 72% 도달 → 난이도 상승 결정,
        //    Environment Manager가 다음 단계 환경으로 전환)
6. decision ← parse_decision(response)
7. cache.store(cache_key, decision)
8. return decision
```

3.4 궤적 평가 및 중요도 가중치 조정

LLM Trajectory Evaluator는 완료된 에피소드의 궤적을 분석하여 학습 가치를 평가한다.

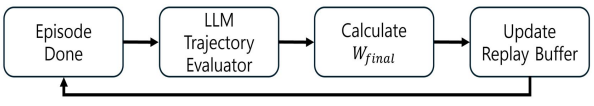

그림 4. 궤적 평가 루프
Fig. 4. Trajectory evaluation loop

그림 4는 궤적 평가 루프의 흐름을 나타낸다. 에피소드가 완료되면 LLM Trajectory Evaluator가 전체 궤적을 프롬프트로 입력받아 관찰값, 행동, 보상의 시퀀스를 종합적으로 평가한다. LLM은 궤적의 학습 가치를 산정하고, 이를 0.1-1.0 범위로 정규화한다. 중요도 가중치(W_{final})는 기존 보상 기반 중요도에 LLM 평가 점수를 가중하여 계산된다.

$$W_{final} = W_{base} + \beta \cdot W_{llm} \quad (1)$$

식 (1)의 W_{base}는 보상 기반 중요도, W_{llm}은 LLM 평가 점수(0.1-1.0로 정규화), β는 LLM의 평가 점수의 반영 비율을 결정하는 고정 가중 계수이다. 계산된 최종 중요도 가중치는 A2C의 경험 재생 버퍼에서 우선순위 가중치로 활용되어 학습 효율성을 높인다. 알고리즘 3은 궤적 평가기의 수도코드이다.

알고리즘 3. 궤적 평가기 수도코드
Algorithm 3. Trajectory evaluator pseudocode

```
Input : trajectory { steps, total_reward, reached_target,
                     collisions, path_deviation,
sharp_turns },
        llm_client, base_importance_weight,
llm_importance_weight
Output: importance_weight (W_final)

1. total_reward ← sum(trajectory.rewards)
2. W_base ← normalize(total_reward)
3. prompt ← format_trajectory_prompt(trajectory)
        // Prompt Template:
        // "Evaluate the learning value of this autonomous driving trajectory.
        //
        // Trajectory Summary:
        //   - Steps : {steps}
        //   - Total reward : {total_reward:.1f}
        //   - Reached target : {reached_target}
        //   - Collisions : {collisions}
        //   - Path deviation : {path_deviation}
        //   - Sharp turns : {sharp_turns}
        //
        // Rate the learning value from 1 to 10,
        // where 10 means highly valuable for training.
        // Respond with only a single integer."
4. response ← llm_client.call(prompt)
        // Output Example: "8"
        //   (학습 가치 높음 → W_llm = 8/10 = 0.8)
5. llm_score ← extract_score(response) or 5.0
6. W_llm ← clamp(llm_score / 10.0, 0.1, 1.0)
7. W_final ← W_base + llm_importance_weight × W_llm
        //   W_final = 0.7 × W_base + 0.3 × 0.8 = 0.7 × W_base + 0.24
8. return W_final
```

IV. 실험 및 평가

4.1 실험 환경 구성

그림 5는 자율주행 시뮬레이션 트랙의 구성을 나타낸다. 실험 환경은 2D 평면 기반 자율주행 시뮬레이션으로 구성하였다. 트랙은 그림 5와 같이 타원형 단일 루프 구조로 구성되며, 내벽과 외벽 사이의 주행 가능 영역에 점선으로 중앙 경로가 표시된다. 차량은 트랙 하단부에 임의의 방향각(yaw)으로 배치되어 에피소드를 시작하며, 목표 지점(TARGET)은 트랙 상단 중앙에 고정 배치된다. 트랙 내에는 복수의 장애물(O)이 배치되며, 에이전트는 이를 회피하면서 목표 지점에 안전하게 도달해야 한다. 환경은 Clear, Light Rain, Heavy Rain, Fog의 4가지 날씨 조건으로 구현하였으며, 동일한 트랙 구조를 기반으로 시야 제한과 차량 제어 난이도가 점진적으로 증가하도록 설계하였다.

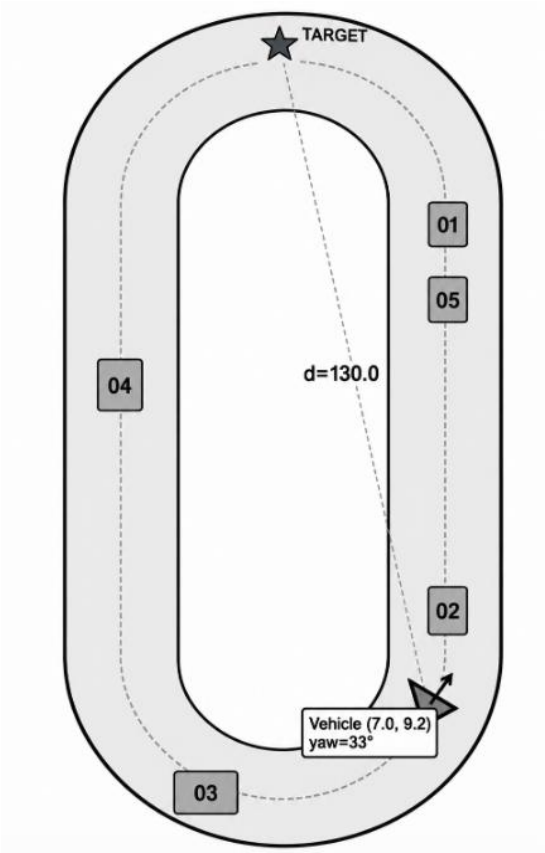

그림 5. 자율주행 시뮬레이션 트랙 구성
Fig. 5. Configuration of the autonomous driving simulation track

에이전트는 4개의 병렬 환경에서 동시에 학습을 수행한다. 보상 함수는 자율주행 차량이 안전하고 효율적으로 목표 지점에 도달하는 것을 주된 목적으로 설정하였다. 에이전트는 기본적으로 경로의 진행도가 높아질 때 양의 보상을 받으며, 경로 이탈, 진행도 감소, 벽 충돌, 급격한 방향 전환이나 비효율적인 경로 선택에 대해서는 음의 보상을 받는다. 각 에피소드는 목표 도달 또는 충돌 시 종료된다. LLM 컴포넌트는 10스텝마다 행동 조언을, 20에피소드마다 난이도 조절 평가를 수행하도록 설정하였다.

4.2 비교 실험 설계

제안 시스템의 효과성을 검증하기 위해 두 가지 모델을 비교 실험하였다. LLM 기반 궤적 중요도 평가와 동적 난이도 조절, 행동 조언 기능을 포함한 LLM-Enhanced A2C 시스템과 전통적인 규칙 기반 접근법을 사용하는 Baseline A2C 시스템이다.

Baseline A2C는 고정된 규칙 기반 커리큘럼 학습을 적용하여 3회 연속 성공 시 다음 난이도로 진행하는 난이도 증가 규칙을 사용한다. 이와 달리 LLM-Enhanced A2C는 성공률, 평균 보상, 에피소드 길이 등의 메트릭을 프롬프트로 LLM에 제공하여 난이도 조절 결정을 수행한다.

비교 대상으로 Baseline A2C만 선정한 이유는 다음과 같다. 이 연구의 핵심 검증 목표는 동일한 A2C 기반 구조에서 LLM 보강 컴포넌트의 순수한 기여 효과를 측정하는 것이다. 따라서 동일 환경에서 재현 가능한 Baseline A2C와의 대조 실험을 통해 LLM 컴포넌트의 독립적 기여도를 측정하는 방식을 선택하였다.

두 시스템 모두 동일한 하이퍼파라미터와 네트워크 구조를 사용하여 비교하였다. 각 시스템은 4개의 날씨 환경에서 250에피소드씩 총 1,000에피소드 동안 학습을 수행하였다.

강화학습에서 보상 신호는 환경 피드백에 직접 기반하므로 W_base는 신뢰도가 높은 지표이다. 반면 LLM 평가(W_llm)는 텍스트 기반 추론에서 비롯되어 환각(hallucination) 가능성이 존재한다. 따라서

보상 기반 중요도에 더 높은 가중치(0.7)를 부여하고 LLM 평가는 보완적 역할(0.3)로 한정하였다.

그 외 실험 환경은 표 1과 같다.

표 1. 실험 환경 구성
Table 1. Experimental environment configuration

Operating system	Ubuntu 24.04
Framework	Python 3.12
Library	PyTorch 2.4.0+ROCm6.3.4
CPU	Ryzen 9 7900X3D
GPU	AMD Radeon RX 7900XTX
Memory	32 GB
LLM	Claude 4 Sonnet
LLM difficulty interval	20 episodes
LLM action interval	10 steps
Episode per environments	250
Base importance	0.7
LLM importance weight	0.3

4.3 날씨별 성능 평가

표 2와 그림 6은 4가지 날씨 조건에서 테스트 이후 정량적 성능 비교 결과를 나타낸다. 테스트는 날씨 조건당 250에피소드를 수행하였다. 모든 날씨 조건에서 LLM-Enhanced A2C가 Baseline A2C 대비 개선된 성능을 보였다. 맑은 날씨 조건에서 LLM-Enhanced A2C는 평균 보상 320.42로 Baseline의 285.02 대비 12.4% 향상된 성능을 보였다. 성공률은 85.2%로 0.6%의 개선되었다.

표 2. 날씨별 성능 비교 결과
Table 2. Performance comparison results by weather conditions

Wheathers	Metrics	LLM-Enhanced A2C	Baseline A2C
Clear weather	Avg reward	320.42	285.02
	Success rate	85.2%	84.6%
Light rain	Avg reward	314.05	261.08
	Success rate	84.8%	80.1%
Heavy rain	Avg reward	298.92	260.05
	Success rate	82.2%	80.0%
Fog	Avg reward	291.70	255.26
	Success rate	82.6%	77.7%

약한 비 조건에서 LLM-Enhanced A2C는 평균 보상 314.05로 20.3%의 성능 향상을 보였으며, 성공률도 84.8%로 4.7% 개선되었다. 폭우 조건에서 LLM-Enhanced A2C는 평균 보상 298.92로 14.9% 향상하였으며, 성공률은 82.2%로 2.2% 개선되었다. 안개 조건에서 LLM-Enhanced A2C는 평균 보상 291.70으로 14.3% 향상을 기록했다. 성공률에서 4.9%의 개선을 보여주었다.

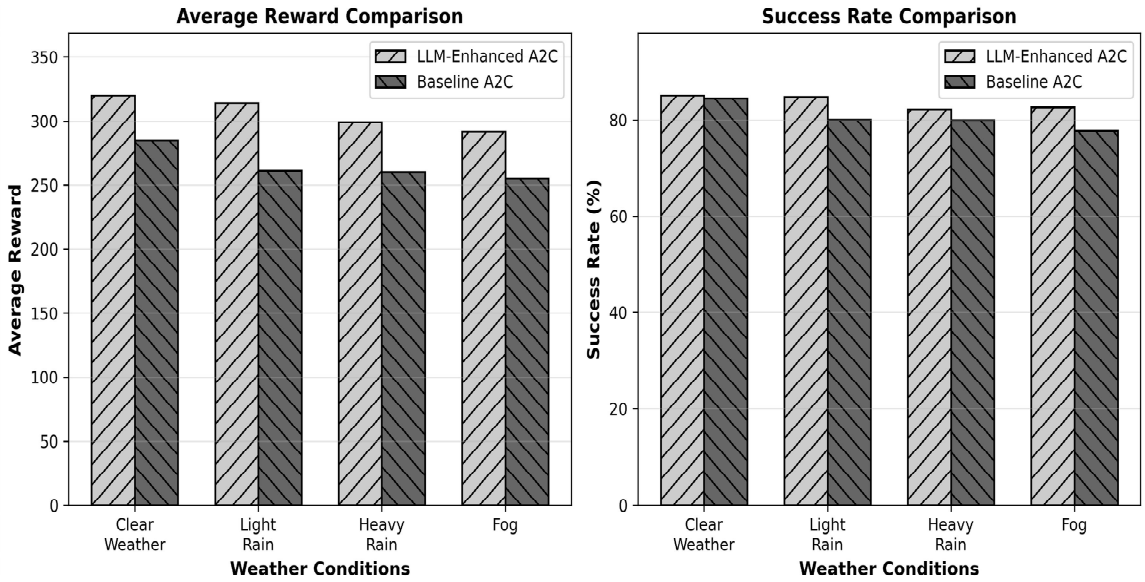

그림 6. 날씨별 성능 비교 결과 그래프
Fig. 6. Performance comparison graphs results by weather conditions

4.4 엔트로피 손실 분석

엔트로피 손실은 값이 0에 가까울수록 에이전트가 최적 정책에 수렴하였음을 나타낸다. 그림 7은 LLM-Enhanced A2C와 Baseline A2C의 학습 과정에서 발생한 엔트로피 손실의 변화를 보여준다. LLM-Enhanced A2C는 학습 후반부에 Baseline A2C 대비 더 빠른 수렴 양상을 보여주었다. 60,000 타임스텝 이후 Baseline A2C의 Entropy가 급격히 감소하는 반면, LLM-Enhanced A2C는 Entropy를 유지하였다.

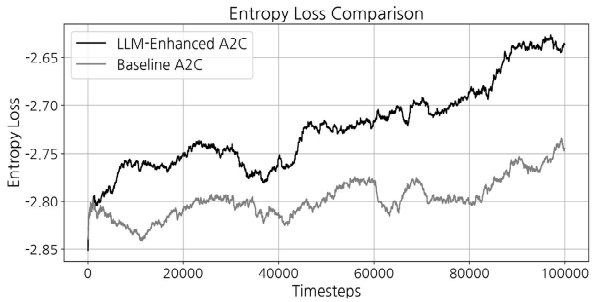

그림 7. 엔트로피 손실 그래프
Fig. 7. Entropy loss graph

V. 결론 및 향후 과제

이 연구에서는 LLM 기반 궤적 중요도 평가를 활용한 A2C 자율주행 시스템을 제안하고, 다중 날씨 환경에서의 성능을 검증하였다. 제안 시스템은 LLM의 상황적 추론 능력과 A2C의 실시간 학습 능력을 통합하여 기존 강화학습 기반 자율주행 시스템의 한계를 극복하고자 하였다. 실험 결과, Entropy Loss에서 LLM-Enhanced A2C는 학습 후반부에 -2.65 수준까지 상승한 반면, Baseline A2C는 학습 후반부에 -2.75 수준까지 감소하였다. 이러한 차이는 LLM 기반 행동 조언과 궤적 중요도 평가가 에이전트의 지속적인 탐험 능력에 도움이 되었음을 의미한다. 날씨별 성능 평가에서도 모든 조건에서 성능 향상이 관찰되었다. Clear Weather 환경에서는 평균 보상이 285.02에서 320.42로 12.4% 향상되었다. Light Rain 환경에서는 가장 큰 성능 향상을 보여 평균 보상이 261.08에서 314.05로 20.3% 증가하였고, 성공률도 80.1%에서 84.8%로 4.7% 개선되었다. Heavy Rain과 Fog 환경에서도 각각 14.9%와 14.3%의 보상 향상을 기록하였다. 안개 환경에서의 성공률 향상은 시야가 제한된 상황에서도 LLM의 도메인 지식과 상황 판단 능력이 성능 개선을 가져올 수 있음을 보여준다. 전체적으로 LLM-Enhanced A2C는 모든 날씨 조건에서 평균 15.5%의 보상 향상과 3.1%의 성공률 개선을 달성했다. 이러한 결과는 LLM의 상황적 추론 능력이 복잡하고 불확실한 환경에서 강화학습 에이전트의 의사결정 품질 향상에 도움이 된다는 것을 보여준다. 실제 자율주행 환경에서 다양한 날씨 조건에 대한 일관성 있는 성능 확보에 의미를 갖는다. 향후 연구에서는 제안한 LLM 기반 궤적 중요도 평가 방법론을 PPO, SAC, TD3 등 다른 강화학습 알고리즘에 적용하여 범용성을 검증할 예정이다. 또한 LLM 행동 조언기, 난이도 조절기, 궤적 평가기 각 컴포넌트의 개별 기여도를 분리 측정하는 ablation study를 수행하고, 바닥의 높낮이 변화와 같은 수직 축(z축)을 포함하는 3D 시뮬레이터(CARLA 등)로의 확장을 통해 실제 자율주행 환경과의 간극을 줄일 예정이다. 아울러 이 연구에서 사용한 외부 LLM API는 호출 시 평균 약 1.2초의 응답 지연이 발생하며, 이는 실제 자율주행 시스템의 실시간 제어 요구사항을 충족하지 못할 수 있다[15]. 이를 해결하기 위해 경량 온디바이스 LLM 또는 사전 계산 방식의 적용을 후속 연구 과제로 추진할 예정이다. 나아가 LLM이 목표 설정, 전략 계획, 자기 반성까지 수행하는 에이전틱 강화학습 구조로의 확장을 통해 LLM이 의사결정의 주체로 기능하는 완전한 LLM 기반 자율주행 시스템을 연구할 예정이다.

References

[1] B. R. Kiran, I. Sobh, V. Talpaert, P. Mannion, A. A. Al Sallab, S. Yogamani, and P. Pérez, "Deep Reinforcement Learning for Autonomous Driving: A Survey", IEEE Transactions on Intelligent Transportation Systems, Vol. 23, No. 6, pp. 4909-4926, Jun. 2022. https://doi.org/10.1109/TITS.2021.3054625.

[2] Z. Yang, X. Jia, H. Li, and J. Yan, "LLM4Drive: A Survey of Large Language Models for Autonomous Driving", arXiv preprint arXiv:2311.01043, pp. 1-19, Aug. 2024. https://doi.org/10.48550/arXiv.2311.01043.

[3] S. Zeng, T. T. Doan, and J. Romberg, "Natural Policy Gradient and Actor Critic Methods for Constrained Multi-Task Reinforcement Learning", arXiv preprint arXiv:2405.02456, pp. 1-42, May 2024. https://doi.org/10.48550/arXiv.2405.02456.

[4] S. Kuutti, R. Bowden, H. Joshi, R. de Temple, and S. Fallah, "End-to-end Reinforcement Learning for Autonomous Longitudinal Control Using Advantage Actor Critic with Temporal Context", Proc. 2019 IEEE Intelligent Transportation Systems Conference (ITSC), Auckland, New Zealand, pp. 2456-2462, Oct. 2019. https://doi.org/10.1109/ITSC.2019.8917387.

[5] M. Park, S. Y. Lee, J. S. Hong, and N. K. Kwon, "Deep Deterministic Policy Gradient-Based Autonomous Driving for Mobile Robots in Sparse Reward Environments", Sensors, Vol. 22, No. 24, pp. 1-18, Dec. 2022. https://doi.org/10.3390/s22249574.

[6] A. Ozturk, M. B. Gunel, R. Dagdanov, M. E. Vural, F. Yurdakul, M. Dal, and N. K. Ure, "Investigating Value of Curriculum Reinforcement Learning in Autonomous Driving Under Diverse Road and Weather Conditions", Proc. 2021 IEEE Intelligent Vehicles Symposium Workshops (IV), Nagoya, Japan, pp. 358-363, Jul. 2021.

[7] S. Jang, H. So, C. Jeong, J. Seo, Y. Hong, S. Kim, E. Shin, and H. Jung, "sA2C-T Reinforcement Learning Method Using Threshold Filtering", Proc. 2024 Korea Information Technology Society Autumn Conference, Jeju, Korea, pp. 999-1003, Nov. 2024.

[8] A. Srinivasan, A. Paras, and A. Bera, "Adversarial Agent Behavior Learning in Autonomous Driving Using Deep Reinforcement Learning", arXiv preprint arXiv:2508.15207, pp. 1-5, Aug. 2025.

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

[10] Y. Wu, D. Li, Y. Chen, R. Jiang, H. P. Zou, W.-C. Huang, Y. Li, L. Fang, Z. Wang, and P. S. Yu, "Multi-Agent Autonomous Driving Systems with Large Language Models: A Survey of Recent Advances, Resources, and Future Directions", Findings of the Association for Computational Linguistics: EMNLP 2025, Suzhou, China, pp. 12756-12773, Nov. 2025.

[11] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized Experience Replay", Proc. International Conference on Learning Representations (ICLR), San Juan, Puerto Rico, pp. 1-21, May 2016. https://doi.org/10.48550/arXiv.1511.05952.

[12] J. Liu, Y. Ma, J. Hao, Y. Hu, Y. Zheng, T. Lv, and C. Fan, "Prioritized Trajectory Replay: A Replay Memory for Data-driven Reinforcement Learning", arXiv preprint arXiv:2306.15503, pp. 1-15, Jun. 2023. https://doi.org/10.48550/arXiv.2306.15503.

[13] Z. Chen, B. Leng, Z. Li, H. Deng, G. Jin, R. Yu, and H. Wen, "HCRMP: A LLM-Hinted Contextual Reinforcement Learning Framework for Autonomous Driving", arXiv preprint arXiv:2505.15793, pp. 1-12, May 2025. https://doi.org/10.48550/arXiv.2505.15793.

[14] J. Wang, A. Karatzoglou, I. Arapakis, and J. M. Jose, "Large Language Model driven Policy Exploration for Recommender Systems", Proc. 18th ACM International Conference on Web Search and Data Mining (WSDM), Hannover, Germany, pp. 107-116, Mar. 2025. https://doi.org/10.1145/3701551.3703573.

[15] S. Park, J. Lee, B.-S. Kim, and S. Jeon, "A

Survey of Proprietary Accelerators for Large Language Models", Journal of Korean Institute of Information Technology, Vol. 23, No. 6, pp. 75-89, Jun. 2025. https://doi.org/10.14801/jkiit.2025.23.6.75.

저자소개

김 동 연 (Dongyeon Kim)

2021년 3월 ~ 현재 :
국립군산대학교 소프트웨어학과
학사과정
관심분야 : LLM, 머신러닝

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