

소화탄 보급을 고려한 강화학습 기반 소방 드론 운용 최적화

2025-2학기 강화학습 이론 및 알고리즘 팀 프로젝트- 12214231 배수찬, 12214260 허재석

1. 문제 선정 동기

① 소방 드론 도입의 가속화

- 최근 산림 화재 현장에서 드론은 단순한 관측을 넘어 작전 수행 도구로서 활용 빈도가 급격히 증가하는 추세 (Ha et al., 2021).
- 특히 접근이 어려운 험준한 지형에서도 효과적으로 작전을 수행할 수 있어, 차세대 핵심 진압 수단으로 주목(Wu et al., 2024).

② 소방 드론 운용의 현실적 난제

- 대형 헬기에 비해 '소화탄 적재량 부족'이라는 물리적 한계가 명확하여 작전 지속성에 제약이 따름 (Kim & Son, 2025).
- 따라서 "제한된 소화탄을 어디에 우선 투하하고, 언제 보급소로 복귀하여 재충전할 것인가?"를 판단하는 운용 효율성 최적화가 중요

③ 강화학습 적용을 통한 의사결정 자동화 제안

- 관측 드론이 파악한 화재 현황(State)을 바탕으로, 타격 드론 군집 (Agent)이 즉각적으로 진압에 나서는 시스템을 가정.
- 강화학습을 통해 얻은 정책으로 소방 드론 시스템을 자동화한다면 보다 즉각적인 대응으로 산불 발생 시 소실 면적을 줄일 것으로 기대

2. 주제 및 문제의 출처

주제: OpenAI Gymnasium 인터페이스 기반의 Grid World 환경에서, 자원 제약과 화재 확산을 동시에 고려한 소방 드론 군집의 최적 진압 경로 학습

환경 출처: Farama Foundation의 MiniGrid 라이브러리를 활용하여 '확률적 화재 확산 모델'과 '보급/재충전 메커니즘'을 포함한 시뮬레이션 환경을 구축하여 이용.

Reference: Github Repository: Farama-Foundation/Minigrid
(Link: <https://github.com/Farama-Foundation/Minigrid>)

3. 문제 정의

시나리오 개요: 24x24 크기의 산림 격자(Grid) 환경에서 화재가 발생하며, 단일 에이전트 (드론 군집)는 화재를 완전히 진압하면서도 피해를 최소화해야 함.

환경의 특징:

- 무작위적 확산: 화재의 발생 시작 위치와 확산은 랜덤하게 이루어짐.
- 자원 제약: 드론은 탑재 가능한 소화탄 개수에 물리적 한계가 있음.
- 보급 메커니즘: 소화탄이 소진되면 보급소로 복귀하여 재장전해야 함.

•하광훈, 김재호, 최재욱 (2021). 소방분야의 드론 활용방안 연구 경향 분석. 한국산학기술학회논문지.
•Wu, R. Y. et al. (2024). *Firefighting Drone Configuration and Scheduling for Wildfire*. Drones.
•Kim, H. & Son, C. (2025). *Research on Multi-Stage Battery Detachment Multirotor UAV to Improve Endurance*. Drones.

환경 설명

MiniGrid환경의 24 X 24 grid world에서 화재의 확산과 드론 군집에 의한 진화 과정을 표현

- 환경 : 자체 제작한 24x24 Grid World 기반의 산불 시뮬레이션 (나무 123개, 장애물 12개로 구성)
- 초기 상태 : 에피소드 시작 시 무작위 3개 지점에서 화재가 발생하여 매 step마다 일정 확률로 상하좌우로 확산
- Agent : (0,0) 충전소에서 보급 후 화재 지점을 방문하여 진압 수행, 모든 화재 진압 또는 수목 전소 시 에피소드 종료

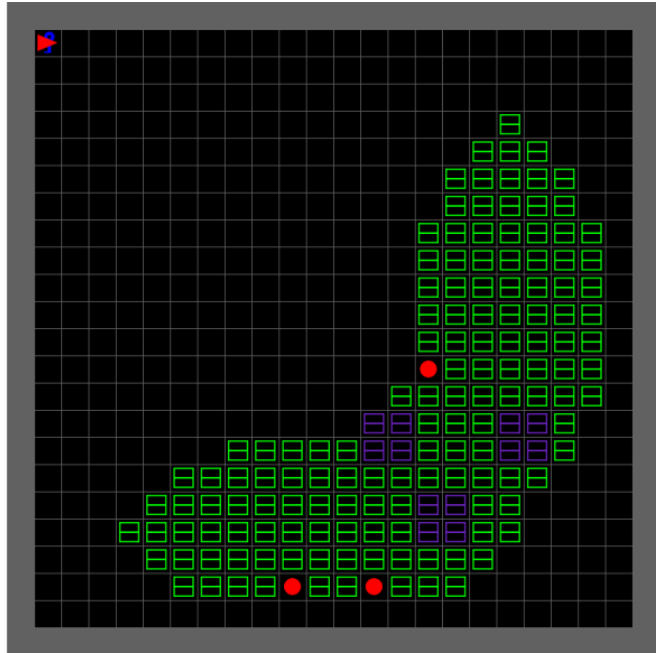
에이전트 제약 설명

1. 배터리 용량 제약 : 제한적인 드론의 비행 시간을 고려하여 드론이 비행하는 최대 step 수는 50으로 제한
2. 최대 이륙 중량 제약 : 드론의 소화탄 적재 용량을 두 개로 제한
3. 운용 효율성 최적화 : 소화탄을 모두 소진하면 즉시 캠프로 복귀하도록 설정해 불필요한 비행을 줄임.






개별 셀의 상태전이 확률 ($S_t = Cell State$)

$Neighbor_{fire}$: 인접한 정상수목이 화재 상태임
 $Visit$: Agent가 해당 격자를 방문함

1. $P(S_{t+1} = Fire | S_t = Healthy, Neighbor_{fire}) = 0.01$: 상하좌우의 셀에 불이 났다면 다음 step에 해당 칸에 불이 옮겨붙을 확률
2. $P(S_{t+1} = Burnt | S_t = Fire) = 0.001$: 이미 불이 붙은 나무가 외부 요인 없이 스스로 소화할 확률
3. $P(S_{t+1} = S_t | S_t \in \{Obstacle, Burnt, Extinguished\}) = 1.0$: 비가연성 지대, 진압완료, 전소 상태에는 불이 옮겨붙지 않는다.
4. $(if I_t > 0) P(S_{t+1} = Extinguished | S_t = Fire, Visit) = 1.0$: 에이전트에게 소화탄이 있다면($I_t > 0$), 화점 방문 시 화재가 진압된다.
5. $(if I_t = 0) P(S_{t+1} = Fire | S_t = Fire, Visit) = 1.0$: 에이전트가 소화탄이 없다면($I_t = 0$), 화점 방문 시 화재가 진압되지 않는다.



〈초기 상태〉

-  정상수목: 연소 가능한 상태의 나무 ($S_t = Healthy$)
-  화재지역: 주변으로 확산 중인 화재 ($S_t = Fire$)
-  비가연성 지대: 화재 확산이 차단되는 장애물
-  진압 완료: 소화 활동으로 보존된 나무 ($S_t = Extinguished$)
-  전소: 화재로 인해 소실된 나무 ($S_t = Burnt$)

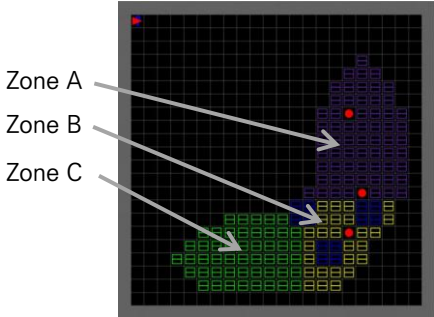
모델 관련 정보 (상태, 액션, 보상 등)

Observation Space

인덱스	변수명	정의 및 설계 의도
1~2	에이전트 절대 좌표 (p_x, p_y)	• 전체 맵(Grid) 내에서의 현재 위치 (0.0~1.0 정규화) : 자신의 구역 위치를 파악하여 이동 경로 계획
3	소화탄 보유 여부 (I_t)	• 소화탄 잔량 비율 (0.0~1.0 정규화, /1: 최대 잔량(2발), 0: 모두 소진) : 화재 지점으로 진입할지, 베이스로 복귀할지 결정하는 기준
4~5	최단 거리 화재 벡터 ($\Delta x_{near}, \Delta y_{near}$)	• 가장 가까운 화재 지점까지의 상대적 거리 (X, Y 축) : 당장 눈앞의 급한 화재를 빠르게 진압하도록 유도
6~7	고위험 화재 벡터 ($\Delta x_{risk}, \Delta y_{risk}$)	• 가장 밀집도가 높은(위험한) 화재 지점까지의 상대 거리 위험도(1~8): 화재발생 지점 주변의 건강한 나무 수에 따라 책정 : 단순 거리보다는 피해가 클 것으로 예상되는 곳을 타겟팅
8~10	구역별 수목 잔존율 (ρ_A, ρ_B, ρ_C)	• 3개 구역(Zone)별 건강한 나무의 비율 : 시야 밖의 숲 전체 상황을 파악하여 거시적 이동 전략 수립

기타

- 의사결정 시점: 매 Time Step (t) 마다 관측 후 행동 수행
 - 할인율: $\gamma = 0.99$
 - 에피소드 종료 조건
1. 모든 화재 진압 완료 ($N_{fire} = 0$)
 2. 숲의 모든 나무가 전소됨 ($N_{healthy} = 0$)
 3. 최대 스텝 도달 ($t \geq 1000$)



Reward

이벤트	보상 값	정의 및 설계 의도
화재 진압	+2.0 ~ + 26.0	화재 진압 지점 주변의 나무가 많을수록 더 큰 보상 부여 기본점수 2.0 + (주변 나무 수) * 3.0
임무 완수	+5 * (남은나무 수)	모든 화재를 성공적으로 진압했을 때 부여되는 보상
임무 실패	-100	모든 나무에 화재가 발생했을 때 부여되는 보상
시간 경과	-0.01/step	에이전트가 불필요한 배회 없이 최단 경로로 이동하도록 유도
충돌 (벽/장애물)	-0.1	맵 밖으로 나가거나 장애물에 부딪히지 않도록 유도
수목 소실	-0.5/tree	최종적으로 최대한 많은 수목을 지키도록 유도
화재 확산	-1.0/tree	

Action Space, 전이확률

에이전트는 매 스텝(t)마다 5가지 이산 행동 중 하나를 선택

$$A_t \in \{0(\text{Stay}), 1(\text{Up}), 2(\text{Right}), 3(\text{Down}), 4(\text{Left})\}$$

$$(\Delta x, \Delta y) = \begin{cases} (0, 0) & \text{if } a_t = 0(\text{Stay}) \\ (0, -1) & \text{if } a_t = 1(\text{Up}) \\ (1, 0) & \text{if } a_t = 2(\text{Right}) \\ (0, 1) & \text{if } a_t = 3(\text{Down}) \\ (-1, 0) & \text{if } a_t = 4(\text{Left}) \end{cases} \quad P(x_{t+1}, y_{t+1} | x_t, y_t, a_t) = \begin{cases} 1 & \text{if } x_{t+1} = x_t + \Delta x, y_{t+1} = y_t + \Delta y \\ 1 & \text{if } x_{t+1} = x_t, y_{t+1} = y_t (\text{Obstacle}) \\ 0 & \text{otherwise} \end{cases}$$

학습 과정, Trial and Error

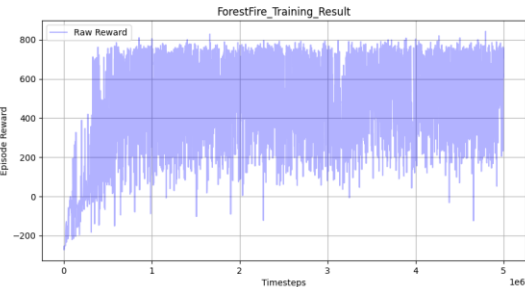
학습 과정

알고리즘: 수업시간에 다룬 PPO 이용하여 학습 (연속형 Space)

Full Code: [GitHub - heozaeseok/minigrid forest RL PPO](#)

rollout/		rollout/	
ep_len_mean	1e+03	ep_len_mean	990
ep_rew_mean	-181	ep_rew_mean	597
time/		time/	
fps	904	fps	894
iterations	3	iterations	2442
time_elapsed	6	time_elapsed	5592
total_timesteps	6144	total_timesteps	5001216
train/		train/	
approx_kl	0.005660612	approx_kl	0.0023677426
clip_fraction	0.013	clip_fraction	0.0205
clip_range	0.2	clip_range	0.2
entropy_loss	-1.59	entropy_loss	-0.202
explained_variance	0.395	explained_variance	0.013
learning_rate	0.0003	learning_rate	0.0003
loss	0.689	loss	175
n_updates	20	n_updates	24410
policy_gradient_loss	-0.00266	policy_gradient_loss	-0.00274
value_loss	3.03	value_loss	702

학습중로그, 학습완료시점로그(500만step학습, 약2시간소요)



LearningCurve, 코드 학습부분 개요

```
from stable_baselines3 import PPO
if __name__ == "__main__":
    # 1. 폴더 생성
    os.makedirs(MODEL_DIR, exist_ok=True)
    os.makedirs(GRAPH_DIR, exist_ok=True)
    os.makedirs(LOG_DIR, exist_ok=True)

    # 2. 환경 생성 및 Monitor 래핑
    # Monitor는 학습 데이터를 csv로 기록해줍니다 (그래프용)
    env = gym.make("ForestFireMLP-v22")
    env = Monitor(env, LOG_DIR)

    print(f"Training Start... (Steps: {TOTAL_TIMESTEPS})")

    # 3. 모델 정의 및 학습
    model = PPO("MlpPolicy", env, verbose=1, device=DEVICE)
    model.learn(total_timesteps=TOTAL_TIMESTEPS)
    print("Training Finished!")
```

Trial and Error

1 : 상태 공간의 한계와 확장

초기 개개의 상태 정보(내 위치, 가장 가까운 화재 등)만으로는 에이전트가 눈앞의 불만 끄는 근시안적 행동을 보임.
(= 숲 전체의 피해상황을 고려하지 못함.)
Solution: 전체 맵을 3개 구역(Zone A, B, C)으로 나누고, 각 구역의 건강한 나무 비율을 상태 값에 추가(7→10차원)
→ 에이전트가 피해가 심각한 구역을 인지하고 거시적인 판단을 할 수 있도록 유도.

2 : 신경망 구조의 전환 (CNN → MLP)

24x24 전체 맵을 이미지로 처리하는 CNN 방식을 시도했으나, 연산 비용이 높고 학습 수렴이 어려웠음.
Solution: 가벼운 MLP Policy로 교체하는 대신, 에이전트가 위치를 인식할 수 있도록 핵심 정보(가장 가까운 화재의 상대 좌표 등)를 직접 가공하여 입력.
→ 핵심 변수만 학습시켜 데이터 효율성 및 수렴 속도 증가

3 : 보상 해킹 방지

화재 확산 페널티가 낮고, 건강한 나무를 많이 구했을 때 보상이 낮아 에이전트가 점수를 더 얻기 위해 불을 끄지 않고 번지게 한 뒤 진입하는 비정상적인 전략을 학습.
Solution: 확산 및 전소 페널티를 강화하고, 완료 보상을 남은 나무 수에 따라 더 받을 수 있도록 상향 조정
→ "피해 최소화 및 신속 진입"이라는 본래의 목적에 부합하도록 보상 체계 재설계.

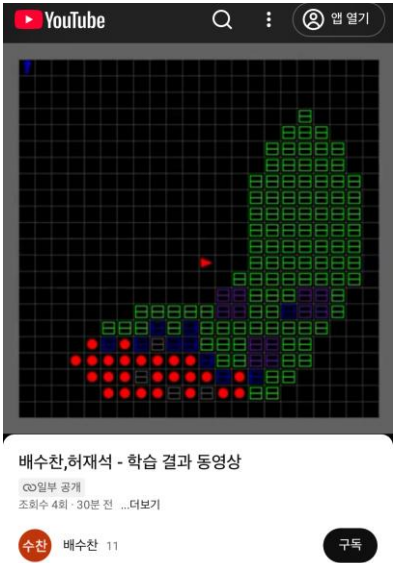
결과 및 한계점

500만 step 학습 결과 동영상

<https://www.youtube.com/watch?v=S8mHmSuxfJw&feature=youtu.be>

Episode 1: 0:00~1:27 / Episode 2: 1:28~3:22

2배속 재생을 권장드립니다.



관찰 및 해석

- 1) 위험구역 우선: 단순히 가까운 불을 끄지 않고, 방치할 경우 확산 피해가 클 것으로 예상되는 위험 구역을 우선 타격하여 연쇄 확산을 차단함.
- 2) 지형지물 활용 및 차단선 구축: 바위나 벽 등 비가연성 장애물을 자연 방화벽으로 활용. 해당 구역은 확산 속도가 느림을 인지하고 진압 순위를 뒤로 미루는 효율적 행동 패턴 관찰.
- 3) 요충지 사수, 선택과 집중: Zone B 화재 시 상하로 불이 번지는 것을 막기 위해 최우선 진압하며, 이미 진압 불가능한 구역은 전체를 과감히 포기하고 살릴 수 있는 숲을 지키는 행동 관찰됨.
- 4) 보상 그래프의 높은 분산 원인: 초기 발화 위치에 따른 난이도 편차가 크기 때문으로 예상됨. 에이전트의 성능 문제보다는 물리적으로 진압이 불가능한 상황들이 포함되어 있어 보상의 변동폭이 크게 발생함.

1. 학습 결과

위험도 기반 진압 전략 수립: 에이전트가 단순히 가까운 화재가 아닌, 확산 확률이 높은 고위험군 화재를 우선 진압하는 전략적 행동 패턴을 학습함.

거시적 상황 판단 능력 확인: 구역별 건강도(Zone Ratio) 정보를 통해, 피해가 심각한 구역으로의 이동 경향성을 보이며 전체 산림 보존율을 향상시킴.

자원 제약 최적화: 배터리(50 step)와 소화제(2 unit)의 물리적 한계 내에서 재보급과 진압 효율을 극대화하는 최적 경로 도출.

2. 한계점

- 1) MLP 기반 모델의 공간 정보 인식 부족으로 인한 비효율적 이동 발생
- 2) 화재 확산과 진압과 관련된 현실적인 요소를 반영하지 못하고 경험에 기반해 확률적으로 접근

3. 향후 연구 방향 제안

- 1) 모델 고도화: 수동 특징 추출의 한계를 넘어, CNN/Attention 도입을 통해 화재 형상 및 지형의 공간적 맥락을 직접 학습하도록 개선
- 2) 환경 정밀도 향상: 풍향, 지형 등 현실의 요소를 더 구체적으로 반영한 시뮬레이션 환경 구축.
- 3) MARL 확장: 다수 드론의 역할 분담 및 협업을 통한 대형 산불 대응 시스템 구축.