드론 GCS 를 위한 LLM 개발

개발 프레임워크 소개


드론 및 디바이스 제어를 위한
Natural Language Command Normalization
LLM Approach
음성명령 → STT → LLM → 명령 제어문 → 드론 제어 → 액션 → 상황 → 추론 → 미션
음성 명령
STT
LLM 쿼리
명령 제어문
드론 액션
피드백 루프

개발 프레임워크 소개

1. 안정성 : 고도 상승 전에 이동하지 않도록 보장 (예: 이륙 → 고도 확보 → 전진).
2. 작전 논리(Mission Logic) : 조건부 순서가 필요함
3. 효율성 : 경로 최적화, 중요임무 우선순위
4. 실시간 적응성 : 예상치 못한 변수에 피드백 루프가 반영되어야 함

연구 내용 소개 개요

1. 관련 연구들 소개
2. LTL
3. PDDL
4. Benchmark
5. 연구방향

2.1 Code as Policies (Liang et al., 2022)

💡 핵심 아이디어

인간이 로봇에게 자연어로 명령하면, 대형 언어 모델이 그 명령을 **실행 가능한 Python 코드**로 변환하는 방법입니다.

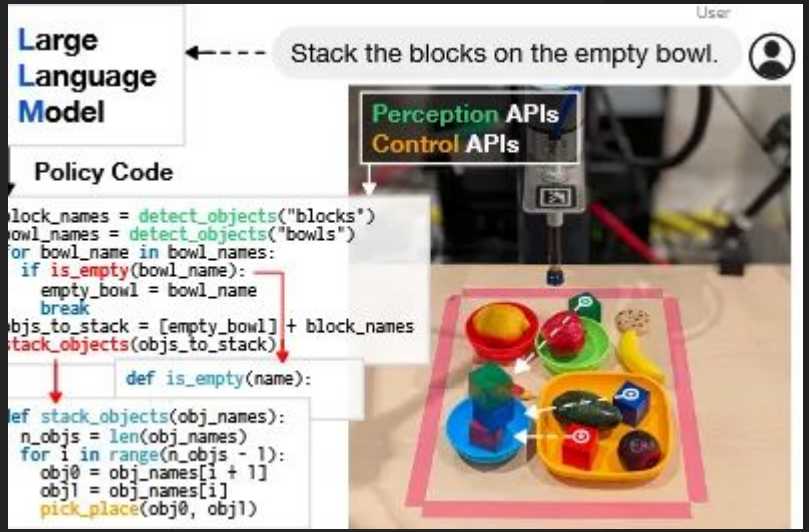
자연어 명령을 로봇 정책 코드로 변환. Few-shot prompting을 통해 예제 명령-코드 쌍을 제공하고, 계층적 코드 생성으로 미정의 함수를 재귀적으로 정의. 제3자 라이브러리(NumPy, Shapely) 활용하여 공간-기하학적 추론 수행.

작동 방식 예시

명령: "빈 그릇에 블록들을 쌓아줘"

생성된 코드:

```
pythonblock_names = detect_objects("blocks")
bowl_names = detect_objects("bowls")
for bowl_name in bowl_names:
    if is_empty(bowl_name):
        empty_bowl = bowl_name
        break
objs_to_stack = [empty_bowl] + block_names
stack_objects(objs_to_stack)
```



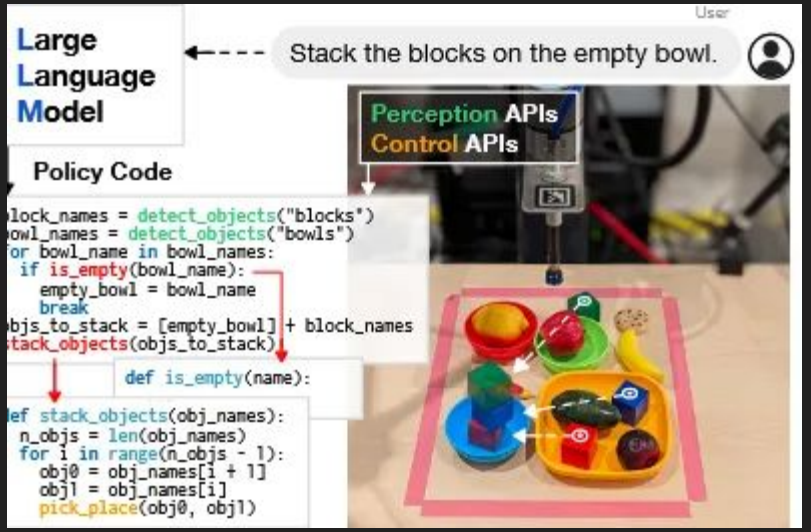
장점

- **유연성:** 새로운 상황에 맞춰 코드를 자동 생성
- **계층적 구조:** 복잡한 함수를 작은 단위로 분해하여 처리
- **실시간 적응:** 환경 변화에 따른 동적 대응 가능

2.1 Code as Policies (Liang et al., 2022)

1. 계층적 Task 방식을 통해 임무 성공률을 높히는 전략이 돋보임
2. 비록 로봇에 도메인 대한 연구 분야지만 적용 가능성을 보았음
3. 퓨샷 프롬프팅으로 코딩 학습이 충분히 가능하다는 가능성

1. 드론 GCS 환경은 코딩위주가 아닌 파싱된 명령어 위주의 미션 플래닝으로 구성됨
2. 퓨샷 프롬프팅 말고 파인튜닝에 집중된 스킬은 돋보이지않음
3. 결론적으로 연구분야로 채택하기 어려움



2.5 Semantic-aware Contrastive Learning (Wu et al., 2022)

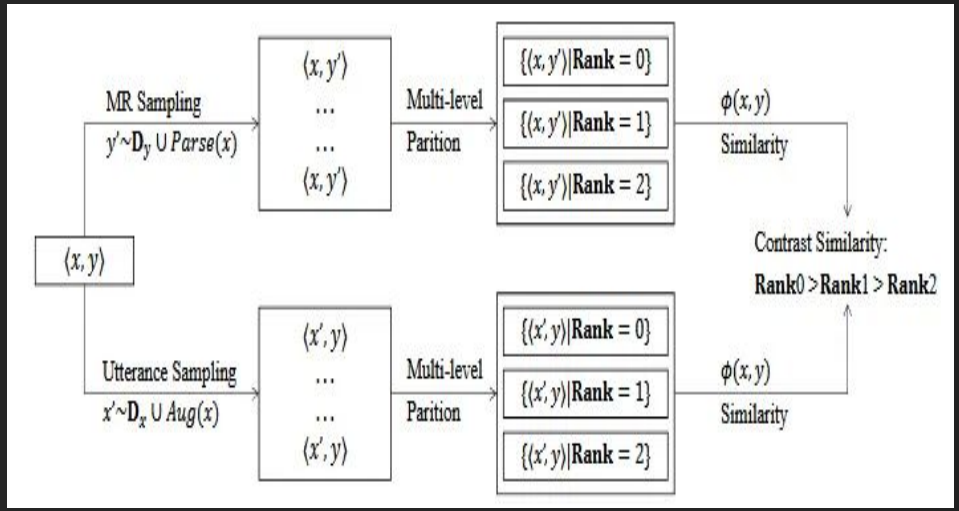
💡 핵심 아이디어

의미가 **미묘하게 다른** 문장들을 정확히 구별할 수 있도록 학습하는 방법입니다.

의미 인식 대조 학습으로 세밀한 의미 표현 학습. 다단계 온라인 샘플링으로 혼동 샘플 수집하고, 3가지 의미 인식 유사도 함수(시퀀스, 어텐션, MR-조건부) 설계. Ranked contrastive loss로 의미적으로 동일한 인스턴스는 가깝게, 다른 인스턴스는 멀게 배치.

문제 상황 예시

- **입력:** "3개 이상 턴오버한 선수"
- **잘못된 변환:** num_turnovers < 3 (3개 미만)
- **올바른 변환:** num_turnovers ≥ 3 (3개 이상)



해결 방법

다단계 샘플링:

- **Rank 0:** 완전히 정답인 예시
- **Rank 1:** 애매한 예시 (패러프레이즈 등)
- **Rank 2:** 완전히 틀린 예시

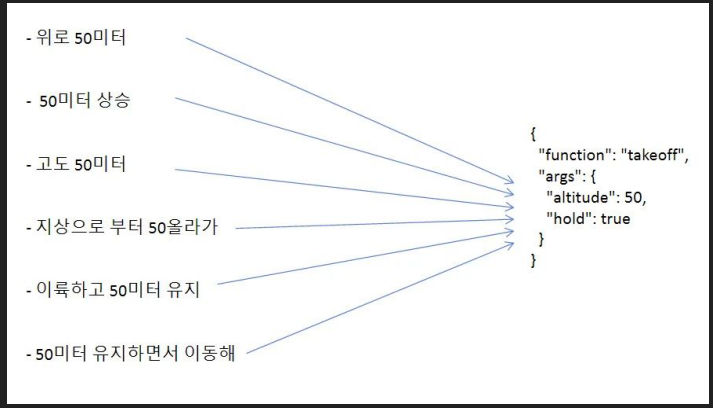
대조 학습: 정답끼리는 가깝게, 틀린 답과는 멀게 배치

실제 효과

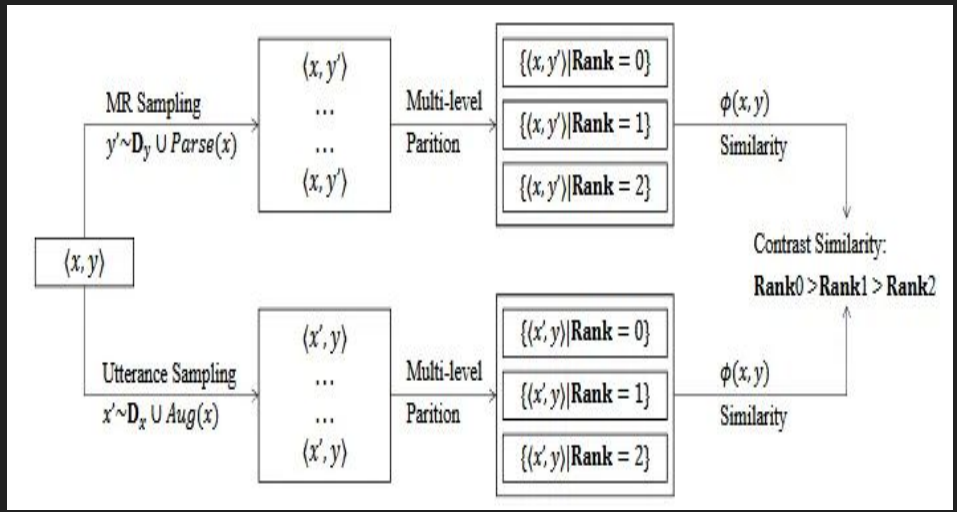
42.7%의 오류가 단 **1글자 차이**로 발생했던 문제를 해결하여, 미세한 의미 차이도 정확히 구별할 수 있게 되었습니다.

2.5 Semantic-aware Contrastive Learning (Wu et al., 2022)

1. 해당 논문의 경우 다양한 한국어 표현에 맞는 명령어 매핑의 중요성을 위해서 찾아보게 되었음
2. 3가지 의미 유사도 함수를 통해서 랭킹하는 전략은 좋았음



1. 실제 언어모델로 작은 실험을 해보니 표현을 바꾸어 설명해도
2. 나름 언어모델이 잘 커맨드를 내뱉는 것을 보았음
3. 자연어 처리 연구 분야와 조금 벗어나는 느낌
4. 결론적으로 연구분야로 채택하기 어려움



2.4 SayCan (Ahn et al., 2022)

💡 핵심 아이디어

대형 언어모델의 **지식**과 로봇의 **실제 능력**을 결합하여 현실적인 계획을 수립하는 방법입니다.

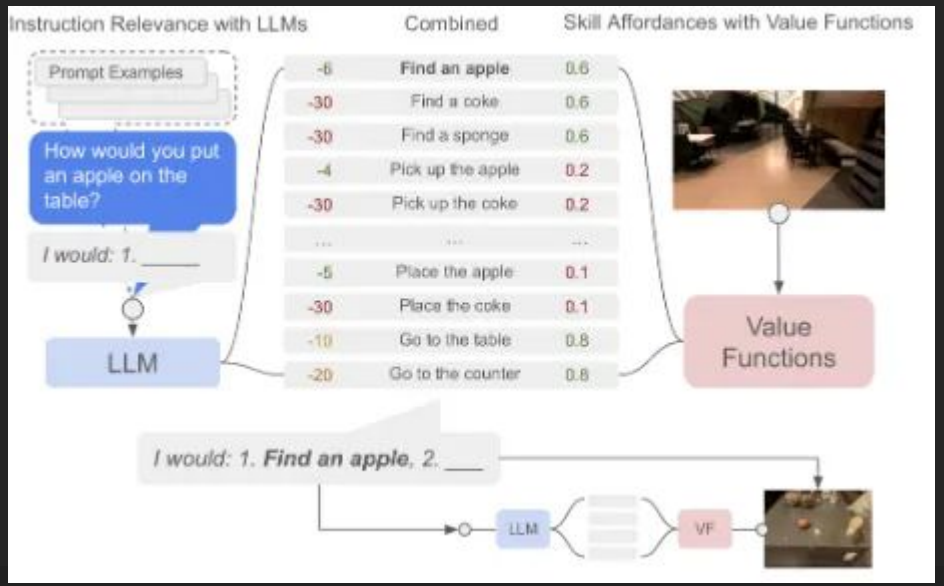
LLM의 의미적 지식과 강화학습으로 학습된 스킬의 affordance 함수를 결합. P(완료|상태,스킬) × P(스킬|명령) 형태로 확률 계산하여 가장 적절한 스킬 선택. 멀티태스크 행동복제(BC-Z)와 강화학습(MT-Opt)으로 언어조건부 정책 학습.

작동 방식 예시

명령: "콜라 한 잔 가져다줘"

단계별 계산:

1. **언어모델 점수:** "콜라 찾기" = 0.8, "콜라 집기" = 0.7
2. **능력 점수:** "콜라 찾기" = 0.9, "콜라 집기" = 0.6
3. **최종 점수:** 0.8 × 0.9 = 0.72 vs 0.7 × 0.6 = 0.42
4. **선택:** "콜라 찾기" 먼저 실행



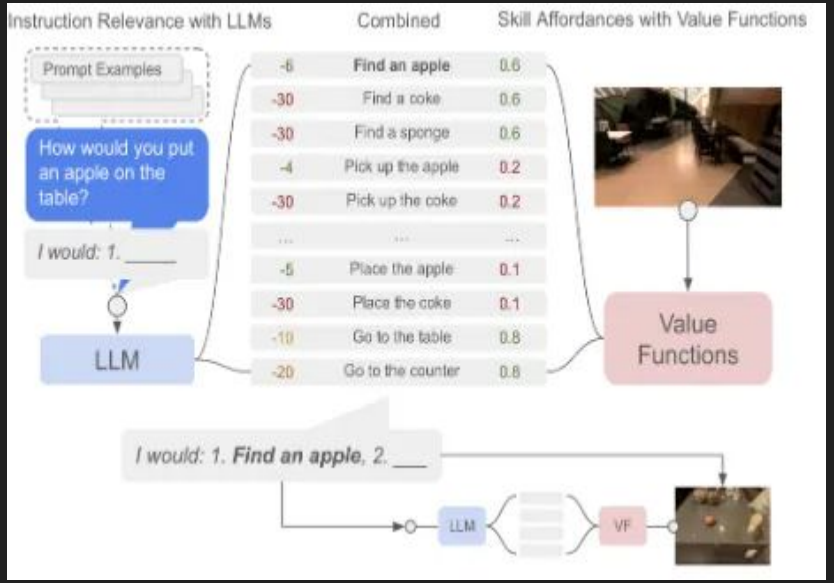
핵심 공식

P(성공) = P(언어적으로 적절함) × P(물리적으로 가능함)

2.4 SayCan (Ahn et al., 2022)

1. 성공확률이 높은 우선순위 선정을 위한 방법으로 적합

1. 행동가능성 확률은 위에서 언급된 행동복제, 강화학습의 대규모 로봇 특화 모델을 베이스로 하여 만드는것임
2. 고로 드론 환경에 특화된 내용이 아니므로 구현하거나 제작하기에 많은 리소스가 듬
2.



2.2 ConformalNL2LTL (Wang et al., 2025)

핵심 아이디어

자연어 명령을 **선형 시간 논리(LTL)** 공식으로 변환하되, **사용자가 지정한 정확도를 보장**하는 방법입니다.

자연어를 LTL 공식으로 변환하는 과정을 연속적인 질문-답변(QA) 작업으로 분해. Conformal Prediction을 활용해 LLM의 불확실성을 정량화하고, 신뢰도가 낮을 때 인간 전문가의 도움을 요청. 다중 응답 샘플링으로 주파수 기반 신뢰도 측정.

작동 방식 예시

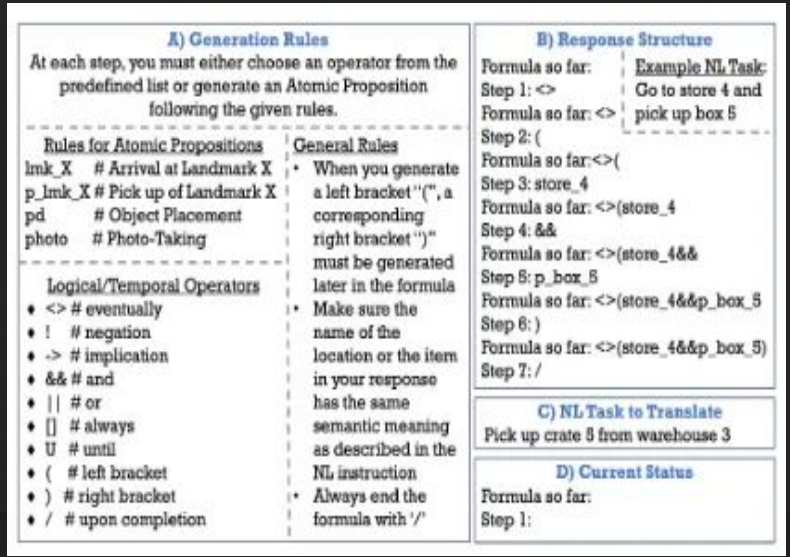
명령: "빨간 상자를 집어서 창고에 놓아줘"

단계별 변환:

1. **질문-응답 방식:** "먼저 무엇을 해야 하나?" → "pick up"
2. **불확실성 측정:** 모델이 여러 답변을 생성해 신뢰도 계산
3. **최종 LTL:** ◇(p_red_box ∧ ◇(storage ∧ pd))

안전장치

- **높은 불확실성** 감지 시 → 사람에게 도움 요청
- **95% 정확도** 보장 → 5% 오류율 허용 범위 내



A) Generation Rules

At each step, you must either choose an operator from the predefined list or generate an Atomic Proposition following the given rules.

Rules for Atomic Propositions

lmk_X # Arrival at Landmark X
p_lmk_X # Pick up of Landmark X
pd # Object Placement
photo # Photo-Taking

Logical/Temporal Operators

- <> # eventually
- ! # negation
- -> # implication
- && # and
- || # or
- [] # always
- U # until
- (# left bracket
-) # right bracket
- / # upon completion

General Rules

- When you generate a left bracket "(", a corresponding right bracket ")" must be generated later in the formula
- Make sure the name of the location or the item in your response has the same semantic meaning as described in the NL instruction
- Always end the formula with '/'

B) Response Structure

Formula so far:
Step 1: <>
Formula so far: <>
Step 2: (
Formula so far:<>(
Step 3: store_4
Formula so far: <>(store_4
Step 4: &&
Formula so far: <>(store_4&&
Step 5: p_box_5
Formula so far: <>(store_4&&p_box_5
Step 6:)
Formula so far: <>(store_4&&p_box_5)
Step 7: /

Example NL Task:
Go to store 4 and pick up box 5

C) NL Task to Translate

Pick up crate 5 from warehouse 3

D) Current Status

Formula so far:
Step 1:

2.3 NL2LTL Package (Fuggitti & Chakraborti, 2023)

💡 핵심 아이디어

비전문가도 **자연어만으로** 복잡한 워크플로우 규칙을 만들 수 있게 해주는 Python 패키지입니다.

자연어 명령을 LTL 공식으로 변환하는 Python 패키지. DECLARE 템플릿 기반 패턴 매칭과 Rasa NLU 엔진 또는 OpenAI API 기반 언어모델 추출 방식 제공. 확장 가능한 템플릿 시스템과 필터링 기능으로 후처리.

사용 예시

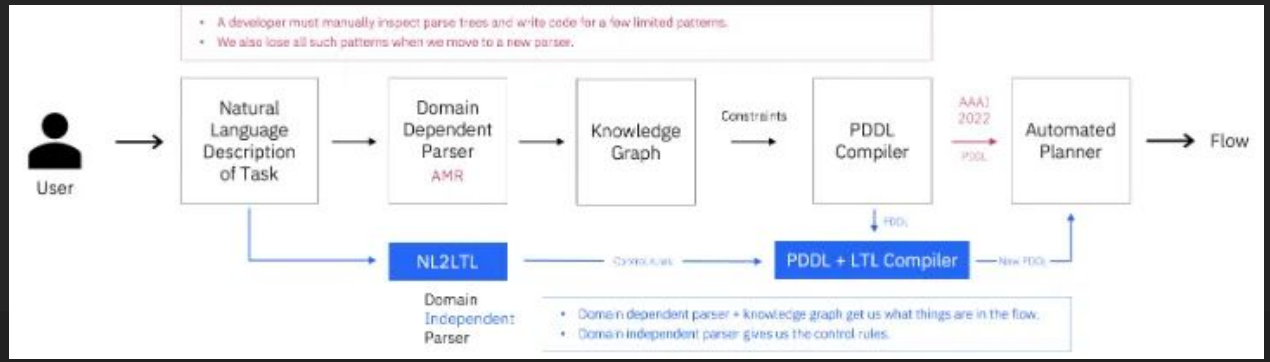
입력: "슬랙 메시지를 받으면 바로 지메일로 답장해야 해"

출력:

- **DECLARE 템플릿:** ChainResponse(Slack, Gmail)
- **영어 설명:** "Slack이 발생할 때마다 Gmail이 바로 따라와야 합니다"
- **신뢰도:** 99.99%

확장성

- 다양한 도메인에 적용 가능
- 새로운 규칙 패턴 추가 용이
- 다국어 지원



LTL (Linear Temporal Logic)

선형 시간 논리

시간에 따라 변화하는 시스템의 동작을 형식적으로 표현하고 **검증**하기 위한 **논리적 언어**입니다. 주로 **형식 검증(formal verification)**, **로봇 경로 계획**, **프로그램 검증**, **자율 시스템 제어** 등에서 사용

계획을 논리적, 시간적, 안전하게 제약하고 표현

1. **LTL**은 시간적으로 변화하는 시스템의 조건을 논리적으로 표현하기 위한 체계
2. **"언제", "항상", "결국"** 같은 시간적 조건을 논리로 표현
3. 로봇, 자율시스템, 임베디드 시스템, 프로그램 검증 등에서 사용됨

왜 중요한가

1. **LLM의 “모호한 명령어”를 “명확한 행동 제약”으로 변환 가능**
2. **드론의 행동을 “시간적으로 제약”하고 “검증 가능하게” 만들 수 있음**
3. **드론 미션 플래닝시 필수적으로 지켜져야할 순서들을 지킴으로서 미션성공률 상승**

자연어 명령	LTL 공식
"이륙 전 시동 켜야 함"	`G(takeoff → engine_on)`
"비행 중 통신 유지"	`G(in_flight → connected)`
"저전력 → 충전해야"	`G(battery_low → F charging)`
"정찰 후 복귀"	`G(mission_complete → F at_base)`
"비행 중 카메라 항상 ON"	`G(in_flight → camera_on)`

G : Globally / 항상
F : Finally / 언젠가

PDDL (Planning Domain Definition Language)

(초기 상태 > 목표 상태) 에 도달하기 위한 “Sequence Of Action” 을 찾는것

계획 문제를 컴퓨터가 이해할 수 있게 표현하는 표준 언어

※ 계획 수립

(초기 상태 → 목표 상태)에 도달하기 위한 "Sequence Of Action"을 찾기 위해서 필요한 여러가지 환경/요소 등을 정의하는 것

>> 이렇게 도출된 Solution Plan은 "Optimal"해야 한다

최소 비용으로 목표에 도달

도메인(domain)

- 가능한 행동(actions), 그 행동의 전제조건(preconditions)과 결과효과(effects)를 정의
- 예: "문을 여는 행동"의 조건과 결과

문제(problem)

- 초기 상태(initial state)
- 달성해야 할 목표(goal)

1. 도메인 정의 (domain.pddl)

lisp

```
(define (domain drone-domain)
  (:requirements :strips :typing)
  (:types drone location)

  ;; 드론이 위치를 이동하는 행동
  (:action fly
    :parameters (?d - drone ?from - location ?to - location)
    :precondition (and (at ?d ?from))
    :effect (and
              (not (at ?d ?from))
              (at ?d ?to)
            )
  )

  ;; 드론이 사진 촬영하는 행동
  (:action take-photo
    :parameters (?d - drone ?loc - location)
    :precondition (at ?d ?loc)
    :effect (photo-taken ?loc)
  )
)
```

LTL 과 PDDL

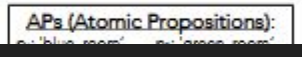
구분	LTL (Linear Temporal Logic)	PDDL (Planning Domain Definition Language)
목적	시스템의 **시간적 행동 제약, 안전성, 순서 조건** 명세	AI 플래너가 이해할 수 있는 **행동과 문제 정의** 언어
용도	**행동 검증, 제약조건 명세, 안전성 보장** 에 주로 사용	**계획 생성** (어떤 행동을 순서대로 해야 목표를 달성하는지 찾기)
표현 방식	논리식으로 “언제, 어떤 조건이 언제까지 참이어야 하는지” 기술	초기 상태, 목표 상태, 행동(전제조건과 효과) 구체적으로 정의
시간 개념	시간의 흐름(미래, 항상, 다음 등)을 논리 연산자로 표현	시간은 명시적으로 표현하지 않고, 행동 순서로 계획 생성
사용 예	안전 조건: "충돌은 절대 발생하지 않아야 한다", "목표를 반드시 거쳐야 한다"	"드론이 어디서 어디로 날아가고, 어떤 행동을 해야 목표 달성하는지"
검증 vs 생성	행동 계획이 규칙에 맞는지 **검증(verification)**에 유리	주어진 도메인에서 목표 달성을 위한 **행동 계획(plan) 생성** 에 특화
복잡성	논리식과 시간 조건 때문에 모델 검증에 강점	실제 문제 해결을 위한 구체적인 행동 설계와 실행에 초점

Benchmark

Data-Efficient Learning of Natural Language to Linear Temporal Logic Translators for Robot Task Specification

Jiayi Pan, Glen Chou, and Dmitry Berenson

Abstract— To make robots accessible to a broad audience, it



로봇 작업 명세를 위한 데이터 효율적인 자연어–LTL 번역기 학습 방법론을 제안

합성 데이터 생성 파이프라인:

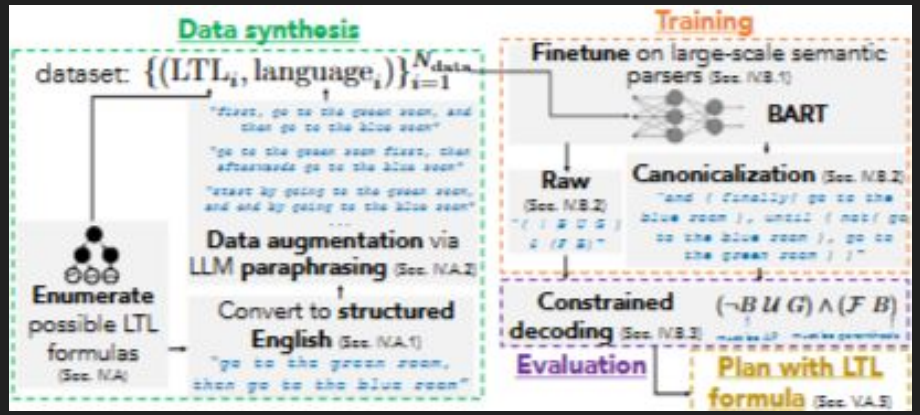
제안: 최소한의 인간 주석(≤12개)만으로 대규모의 다양한 자연어–LTL 훈련 데이터셋을 자동으로 생성하는 파이프라인을 제안

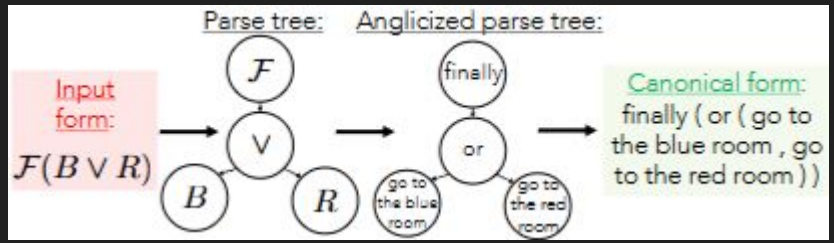
LLM 기반 번역 아키텍처 및 LTL 표현 방식 비교:

제안: 사전 학습된 LLM(BART)을 LTL 번역 태스크에 미세 조정하고, 추론 시 LTL 문법 제약을 강제하는 '제약 조건 디코딩'을 적용하는 아키텍처를 제안
기여: LTL 수식의 두 가지 표현 방식(원시 LTL vs. 자연어에 가까운 Canonical Form)을 LLM의 훈련 레이블로 실험하여, 특정 태스크와 복잡도에 따른 각 방식의 효과를 비교 분석하고 NLP 최신 기술의 LTL 번역 적용 가능성을 입증

Benchmark

- 데이터합성
  - 가능한 LTL 수식열거
    - 모든 가능한 LTL 수식들을 체계적으로 생성
  - 구조화된 영어로 변환
    - 예를 들어, LTL F (green)은 "go to the green room"과 같이 변환
  - LLM 의역을 통한 데이터 증강
    - 한마디로 같은 LTL 수식에 여러가지 문장구조 뜻
    - 합성데이터 순서 LTL 생성 > 자연어 번역 > 여러문장 증강
- 훈련 (Training)
  - 합성된 (LTL_i, language_i) 데이터셋을 사용하여 사전 훈련된 BART와 같은 대규모 언어 모델을 미세 조정
  - 레이블 변형
    - Raw: LTL 수식을 (! B U G) & (F B)와 같은 원시 (raw) 텍스트 형태로 사용
    - Canonicalization: LTL 수식을 자연어에 더 가까운 "정규화된(canonical)" 형태로 변환하여 사용합니다. 예를 들어, F(B ∨ R)은 "finally (or (go to the blue room , go to the red room))"와 같이 변환
- 평가 및 로봇 계획 (Evaluation / Plan with LTL formula)
  - 제약 조건 디코딩
    - 이 과정은 출력 토큰을 미리 정의된 LTL 문법 규칙에 따라 제한하여 잘못된 형식의 LTL 수식이 생성되는 것을 방지
    - 제약조건 해지한 실험도 진행함 ablation
  - LTL 수식을 통한 계획
    - 최종적으로 번역된 LTL 수식은 로봇의 장기적이고 다단계적인 작업을 계획하는 데 사용





Benchmark

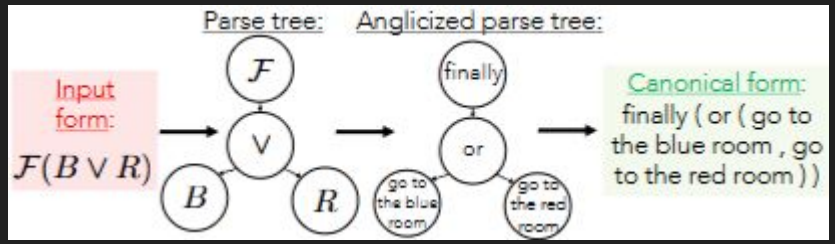
데이터셋 설명

레이블 구성

1. Raw LTL (원시 LTL)
 LTL 연산자(F, G, U, ∧, ∨, ¬)와 원자 명제(Atomic Proposition, p1, blue_room 등)를 기호 그대로 사용하여 표현된 것

2. Canonical Form (정규형 LTL)
구조화된 영어 문장 형태로 변환한 중간 표현
finally (or (go to the blue room , go to the red room))

!분포 차이를 완화하고 학습 효율성을 높이는 용도
! RAW LTL 로 역변환 을 통해 최종 출력

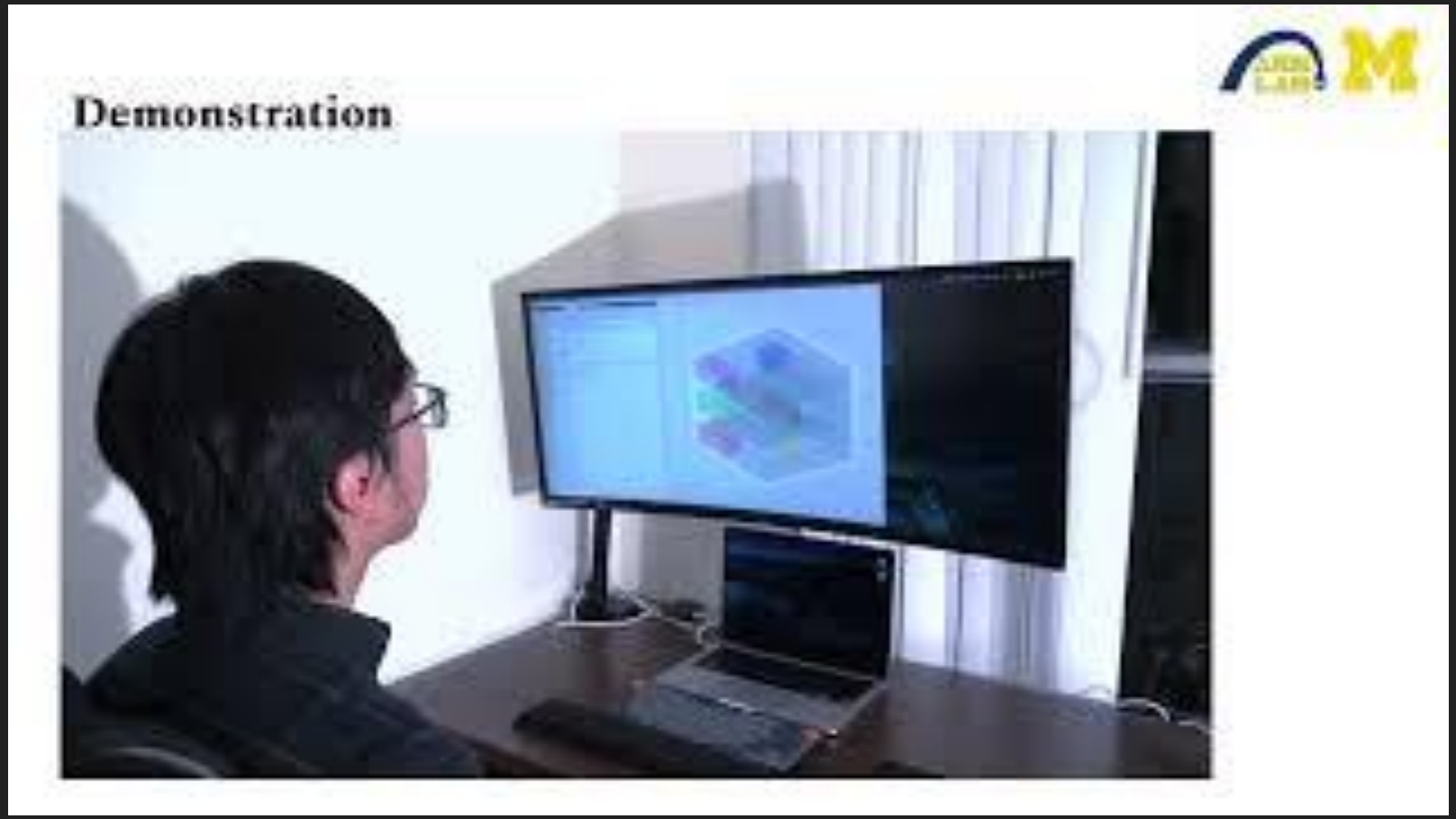


데이터 성격

1. Golden Dataset (골든 데이터셋)
람이 직접 자연어 명령과 해당 LTL 수식을 짝지어 (human-labeled) 주석(annotation)한 고품질의 훈련 또는 평가 데이터셋

2. Synthetic Training Data (합성 훈련 데이터)
대규모 언어 모델(LLM)의 생성 능력을 활용하여 자동으로 생성된 훈련 데이터

Benchmark



Benchmark

Model architecture	Training data	Test data	Drone (5/343)	Cleanup (4/39)	Pick (1/5)
RNN [6]	4/5 golden	1/5 golden	87.18	95.51	93.78
CopyNet [29]	4/5 golden	1/5 golden	88.97	95.47	93.14
BART-FT-Raw (ours)	4/5 golden	1/5 golden	**90.78**	**97.84**	**95.97**
BART-FT-Canonical (ours)	4/5 golden	1/5 golden	90.56	97.81	95.70
RNN [6]	synthetic	full golden	22.41	52.54	32.39
CopyNet [29]	synthetic	full golden	36.41	53.40	40.36
BART-FT-Raw (ours)	synthetic	full golden	**69.39**	**78.00**	**81.45**
BART-FT-Canonical (ours)	synthetic	full golden	68.99	77.90	78.23
BART-FT-Raw-NoConstrainedDecoding	synthetic	full golden	68.23	76.26	81.05
BART-FT-Canonical-NoConstrainedDecoding	synthetic	full golden	67.45	72.06	69.49
BART-FT-Raw (ours)	synthetic; no augmentation	full golden	29.43	52.51	80.38
BART-FT-Canonical (ours)	synthetic; no augmentation	full golden	39.21	53.16	67.88

모델 아키텍처	훈련 데이터	테스트 데이터	드론 (5/343)	클린업 (4/39)	픽 (1/5)
RNN [6]	golden 데이터 4/5	golden 데이터 1/5	87.18	95.51	93.78
CopyNet [29]	golden 데이터 4/5	golden 데이터 1/5	88.97	95.47	93.14
BART-FT-Raw (본 논문)	golden 데이터 4/5	golden 데이터 1/5	90.78	97.84	95.97
BART-FT-Canonical (본 논문)	golden 데이터 4/5	golden 데이터 1/5	90.56	97.81	95.70
RNN [6]	합성	전체 golden 데이터	22.41	52.54	32.39
CopyNet [29]	합성	전체 golden 데이터	36.41	53.40	40.36
BART-FT-Raw (본 논문)	합성	전체 golden 데이터	69.39	78.00	81.45
BART-FT-Canonical (본 논문)	합성	전체 golden 데이터	68.99	77.90	78.23
BART-FT-Raw-NoConstrainedDecoding	합성	전체 golden 데이터	68.23	76.26	81.05
BART-FT-Canonical-NoConstrainedDecoding	합성	전체 golden 데이터	67.45	72.06	69.49
BART-FT-Raw (본 논문) 합성; 증강 없음	합성; 증강 없음	전체 golden 데이터	29.43	52.51	80.38
BART-FT-Canonical (본 논문) 합성; 증강 없음	합성; 증강 없음	전체 golden 데이터	39.21	53.16	67.88

결론 :

정제된 golden 데이터셋이 성능면에서 월등히 우수함

canonical 의 실험결과가 두드러지지 않는데 저자들은 이 부분에 대해 캐노니컬 형태의 이점을 충분히 활용할 만큼 충분히 복잡하지 않았을 가능성을 제기

또한 데이터 증강의 중요성은 표를 통해 증명됨 69 > 29

연구 방향

1. **다국어 도메인 확장: 한국어 LTL 번역기 개발**
   a. 데이터 합성 파이프라인 현지화 및 문화적/언어적 특성 반영

2. **데이터 증강 없는 효율적인 학습 방법 연구: LoRA (Low-Rank Adaptation) 등 활용**
   a. 데이터 증강의 양과 품질을 최적화하여 최소한의 증강으로도 충분한 성능을 달성할 수 있는 전략을 모색
   b.

3. **제약 조건 디코딩(Constrained Decoding) 보다 더 나은 방법을 찾아서 성능을 개선**
   a. 제약 조건 디코딩은 실험결과에서 큰 이점을 보이지않았음
   b.

4. **드론 제어 명령 LLM을 개발하면서 LTL 제약식 개념을 추가하는 방식**
   a. 한번에 모든 미션을 출력하는 LLM 형태 > 드론 행동 오류 발생
   b. 제약식 조건으로 우선시 되는 행동 및 A 행동 다음 B 행동 원칙