



Taking Flight with Dialogue: Enabling Natural Language Control for PX4- based Drone Agent

📖 Notebooks	📁 <u>DJI GCS project</u>
🗄 보관함	□
■ 상태	작성 중
■ 생성 일시	@2025년 8월 8일 오전 6:00
★ 즐겨찾기	□
■ 최종 편집 일시	@2025년 8월 8일 오후 3:18


Taking Flight with Dialogue: Enabling Natural Language Control for PX4-based Drone Agent

Shoon Kit Lim 


*University of Southampton Malaysia,
Iskandar Puteri, Johor, Malaysia
skl1g14@soton.ac.uk*

Melissa Jia Ying Chong 

*University of Southampton Malaysia,
Iskandar Puteri, Johor, Malaysia
m.j.y.chong@soton.ac.uk*

Jing Huey Khor 

*Connected Intelligence Research Group,
University of Southampton Malaysia,
Iskandar Puteri, Johor, Malaysia
j.khor@soton.ac.uk*

Ting Yang Ling 

*Sustainable Electronic Technologies,
University of Southampton,
Southampton SO17 1BJ U.K.
ivan.ling@soton.ac.uk*

PX4를 기반으로 하는 드론 에이전트(Drone Agent)

PX4는 드론의 비행 제어를 담당하는 오픈 소스 자동조종 장치이며,

"Drone Agent"는 단순한 로봇이 아닌 자율적으로 판단하고 행동할 수 있는 지능형 드론 시스템을 의미합니다.

이 논문은 ROS2 미들웨어와 Ollama를 사용하여 PX4 기반 드론에 LLM과 VLM을 통합함으로써, 드론이 복잡한 환경에서 자연어 명령에 따라 자율적으로 임무를 수행하도록 하는 프레임워크를 제시합니다.

요약

• 연구 배경 및 문제점:

- 최근 에이전트 및 물리적 AI 기술 발전은 주로 휴머노이드, 바퀴 달린 로봇과 같은 지상 기반 플랫폼에 집중되어 공중 로봇(드론) 분야는 상대적으로 덜 탐구되었습니다.
- 최첨단 UAV 멀티모달 시각-언어 시스템은 일반적으로 자원(재정, 인력 등)이 풍부한 기관에서만 접근 가능한 클로즈드 소스 모델에 의존합니다. 이는 드론의 자연어 제어 기술의 대중화를 저해합니다.

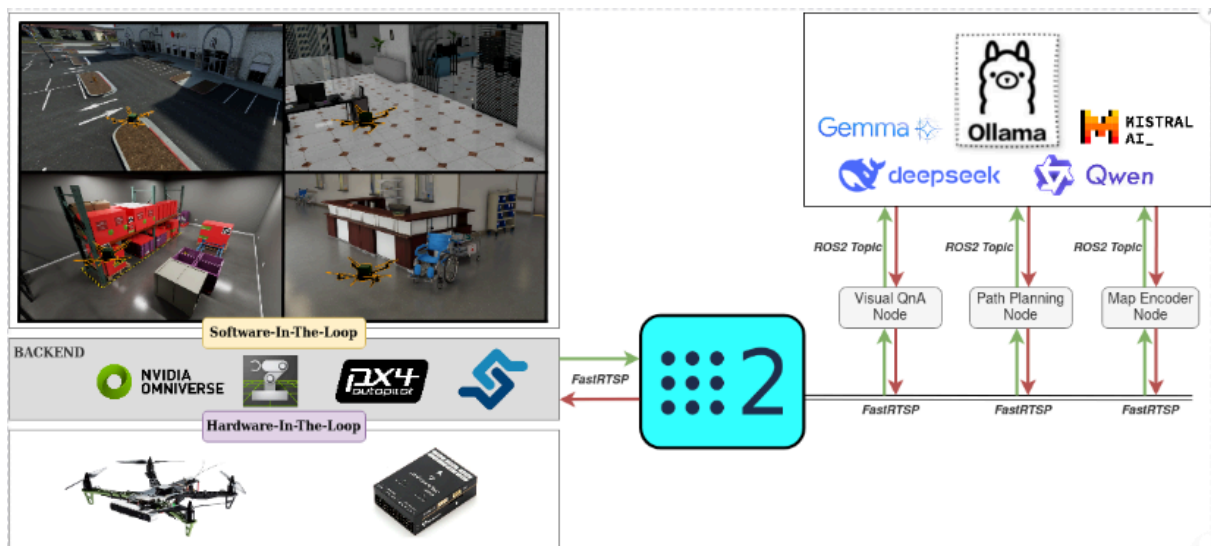
• 제안하는 솔루션:

- 자율 드론의 자연어 제어를 대중화하기 위해, PX4 기반 비행 제어, ROS2 미들웨어, 그리고 Ollama를 사용하여 로컬에서 호스팅되는 모델들을 통합한 오픈 소스 에이전트 프레임워크를 제시합니다.

• 성능 평가:

- 시뮬레이션 환경과 맞춤형 쿼드콥터 플랫폼 모두에서 성능을 평가했습니다.
- **LLM(대규모 언어 모델) 벤치마킹:** 명령 생성을 위해 네 가지 LLM 제품군 (Gemma3, Qwen2.5, Llama-3.2, DeepSeek-LLM)을 비교했습니다.

- Gemma3, Qwen2.5, Llama-3.2는 일관되게 100% 유효한 비행 명령을 생성했습니다.
- DeepSeek-LLM은 38%로 현저히 낮은 성능을 보였습니다.
- **VLM(시각-언어 모델) 벤치마킹:** 장면 이해를 위해 세 가지 VLM 제품군 (Gemma3, Llama3.2-Vision, Llava1.6)을 비교했습니다.
 - 평가된 모든 VLM(Gemma3, Llama3.2-Vision, Llava1.6)은 지정된 객체 유무를 감지하고 97%에서 100% 범위의 유효한 이진 응답(예/아니오)을 제공할 수 있었습니다.
- **임무 성공률:** 모델 조합에 따라 임무 성공률이 다양하게 나타났습니다.
 - Gemma3 LLM과 VLM 조합에서 40%로 가장 높은 성공률을 기록했습니다.



제안하는 프레임워크

• **Software-In-The-Loop (SITL) 시뮬레이션 환경:**

- 이 부분은 NVIDIA Omniverse와 Isaac Sim, 그리고 PX4 Autopilot을 사용하여 가상 환경에서 드론 비행을 시뮬레이션합니다.

- 이미지 상단에는 야외 주차장, 코워킹 스페이스, 병원, 창고, 데이터 센터 등 다양한 시뮬레이션 환경이 예시로 제시되어 있습니다.
- SITL은 실제 드론을 사용하기 전에 다양한 시나리오에서 시스템의 성능을 테스트하고 검증하는 데 사용됩니다.
- **Hardware-In-The-Loop (HITL) 실제 드론 플랫폼:**
 - 이 부분은 실제 맞춤형 쿼드콥터와 그에 탑재된 Jetson Orin Nano Dev Kit, ZED Mini 카메라, Pixhawk 6c Mini 비행 컨트롤러와 같은 하드웨어 구성 요소를 나타냅니다.
 - HITL은 시뮬레이션에서 검증된 시스템을 실제 드론에 적용하여 실제 환경에서의 성능을 평가합니다.
- **ROS2 (Robot Operating System 2) 미들웨어:**
 - ROS2는 프레임워크의 핵심적인 통합 플랫폼으로, 시스템의 다양한 모듈 간의 통신을 담당합니다.
 - 시뮬레이션 또는 실제 드론으로부터 비주얼 데이터와 상태 정보가 FastRTSP 프로토콜을 통해 ROS2로 전송됩니다.
- **Ollama 및 대규모 언어/시각-언어 모델 (LLM/VLM):**
 - Ollama는 Gemma, DeepSeek, Mistral AI, Qwen, Llama 등 다양한 LLM 및 VLM을 로컬에서 호스팅하고 서빙하는 플랫폼입니다.
 - 이 모델들은 ROS2 토픽을 통해 다양한 노드와 상호작용합니다.
- **모듈형 노드:**
 - **Visual QnA Node:** VLM (Gemma, Llama3.2-Vision, Llava1.6 등)을 사용하여 드론의 현재 시각적 관찰(이미지)과 사용자 쿼리를 처리하여 텍스트 응답(예: "객체가 있습니까?")을 생성합니다. 이는 드론이 환경을 이해하는 데 도움을 줍니다.
 - **Path Planning Node:** LLM을 사용하여 목표 지점과 드론의 현재 자세를 PX4 비행 동작으로 변환하여 충돌 없는 궤적을 계획합니다. 이는 드론이 목적지까지 안전하게 이동할 수 있도록 합니다.
 - **Map Encoder Node:** 드론의 현재 자세와 이미지에서 얻은 의미론적 정보를 텍스트 토큰으로 임베딩하여 지도를 표현합니다. 이는 드론이 환경에 대한 내부적인 이해를 구축하는 데 기여합니다.
- **작동 방식:**

- 드론(시뮬레이션 또는 실제)은 카메라를 통해 시각 데이터를 획득하고, 이 데이터는 ROS2를 통해 **Visual QnA Node** 로 전달됩니다.
- 사용자의 자연어 명령과 시각적 정보, 그리고 드론의 현재 상태 및 이전 동작 이력 등은 LLM에 입력되어 **Path Planning Node** 를 통해 적절한 비행 명령(예: **Turn(θ)** 또는 **Move(d)**)으로 변환됩니다.
- 이 명령은 다시 ROS2를 통해 드론의 PX4 비행 컨트롤러로 전달되어 드론이 자율적으로 비행합니다.
- 이 프레임워크는 오픈 소스 모델과 플랫폼을 활용하여 드론의 자연어 제어를 민주화하는 것을 목표로 합니다. 기존 연구들이 주로 지상 로봇에 집중하고 폐쇄형 모델에 의존했던 것과 달리, 이 연구는 항공 로봇 분야와 오픈 소스 솔루션에 초점을 맞춥니다. RT-1, RT-2와 같은 이전 로봇 학습 프레임워크는 대규모 데이터셋과 트랜스포머 아키텍처를 사용하여 일반화 능력을 향상시켰지만, 이 논문은 이러한 아이디어를 UAV에 적용하고, 특히 Ollama와 같은 로컬 호스팅 가능한 오픈 소스 LLM/VLM에 집중한다는 점에서 차별점을 가집니다.

추가적으로, 이 연구는 SayCan 및 SayTap과 같이 언어 모델을 로봇 동작에 연결하려는 노력을 UAV 분야로 확장합니다. 또한, CityNav나 AerialVLN과 같은 UAV를 위한 기존 데이터셋들이 시뮬레이션 환경에 국한되거나 실세계 검증이 부족하다는 비판을 고려하여, 이 프레임워크는 시뮬레이션과 실제 환경 모두에서 평가를 수행합니다.

실험 환경 구성 및 실험



실험 환경 구조

- **a) 맞춤형 쿼드콥터 (Customized Quadcopter)**
 - 이 그림은 연구에 사용된 실제 쿼드콥터를 보여줍니다.
 - 이 쿼드콥터에는 온보드 계산을 위한 NVIDIA Jetson Orin Nano Dev Kit, 시각-관성 오도메트리(VIO)를 위한 ZED Mini 카메라, 그리고 모터 제어를 위한 Pixhawk 6c Mini flight controller가 장착되어 있습니다 (3페이지 참조).
 - 이는 본 연구의 핵심인 자연어 제어 드론 에이전트의 실제 플랫폼이 됩니다.
- **b) 쿼드콥터의 디지털 트윈 (Digital Twin of the Quadcopter)**
 - 이 그림은 실제 쿼드콥터의 가상 모델, 즉 디지털 트윈을 보여줍니다.
 - 디지털 트윈은 NVIDIA Isaac Sim 시뮬레이션 환경 내에서 드론의 동작을 정확하게 재현하는 데 사용됩니다.
 - 이를 통해 실제 하드웨어를 사용하기 전에 다양한 비행 시나리오와 알고리즘을 안전하고 효율적으로 테스트할 수 있습니다.
- **c) 실내 비행 환경 (Indoor Flight Environment)**
 - 이 그림은 드론의 실제 비행 테스트가 이루어진 실내 환경을 보여줍니다.
 - 이 공간은 7m x 4.5m x 2.2m 크기의 나일론 그물망으로 둘러싸여 있으며, 내부에 정적 장애물 역할을 하는 판지 상자들이 배치되어 있습니다 (3페이지 참조).
 - 이 환경은 드론의 충돌 회피 및 목표물 탐색 능력을 평가하기 위해 설계되었습니다.
- **d) 환경의 디지털 트윈 (Digital Twin of the Environment)**
 - 이 그림은 실제 실내 비행 환경의 가상 버전, 즉 디지털 트윈을 보여줍니다.
 - 이 디지털 트윈은 Isaac Sim에서 구현되어 소프트웨어-인-더-루프(SITL) 시뮬레이션에 사용됩니다 (2페이지 참조).
 - 시뮬레이션 환경은 실제 환경과 동일하게 장애물을 포함하며, 드론 에이전트의 성능을 가상으로 평가하는 데 필수적인 역할을 합니다.

$$\text{Traj}_{0:K} = (S_0, S_1, \dots, S_K)$$

드론 궤적 정의 수식

- $\text{Traj}_{0:K}$: 드론 에이전트가 시간 0부터 K 까지 이동하는 전체 궤적을 나타냅니다. 궤적은 일련의 상태(state)들로 구성됩니다.
- S_0 : 드론의 알려진 초기 상태(initial state)를 나타냅니다. 이는 드론이 임무를 시작하는 시점의 위치와 자세 등입니다.
- S_1, \dots, S_K : 시간 단계 k 에서의 드론의 상태를 나타냅니다. 각 S_k 는 드론의 위치, 속도, 자세 등 특정 시점의 모든 관련 정보를 포함합니다.
- S_K : 드론이 도달하고자 하는 목표 상태(terminal state)를 나타냅니다. 즉, 임무가 끝나는 시점의 드론 상태입니다.
- K : 전체 궤적의 최종 시간 단계를 의미합니다. 이 값은 에이전트가 임무를 완료하기 위해 필요한 이산 시간 스텝의 총 개수를 나타냅니다.

$$S_{k+1} = \text{PX4}(S_k, A_k)$$

드론 상태 전이 공식

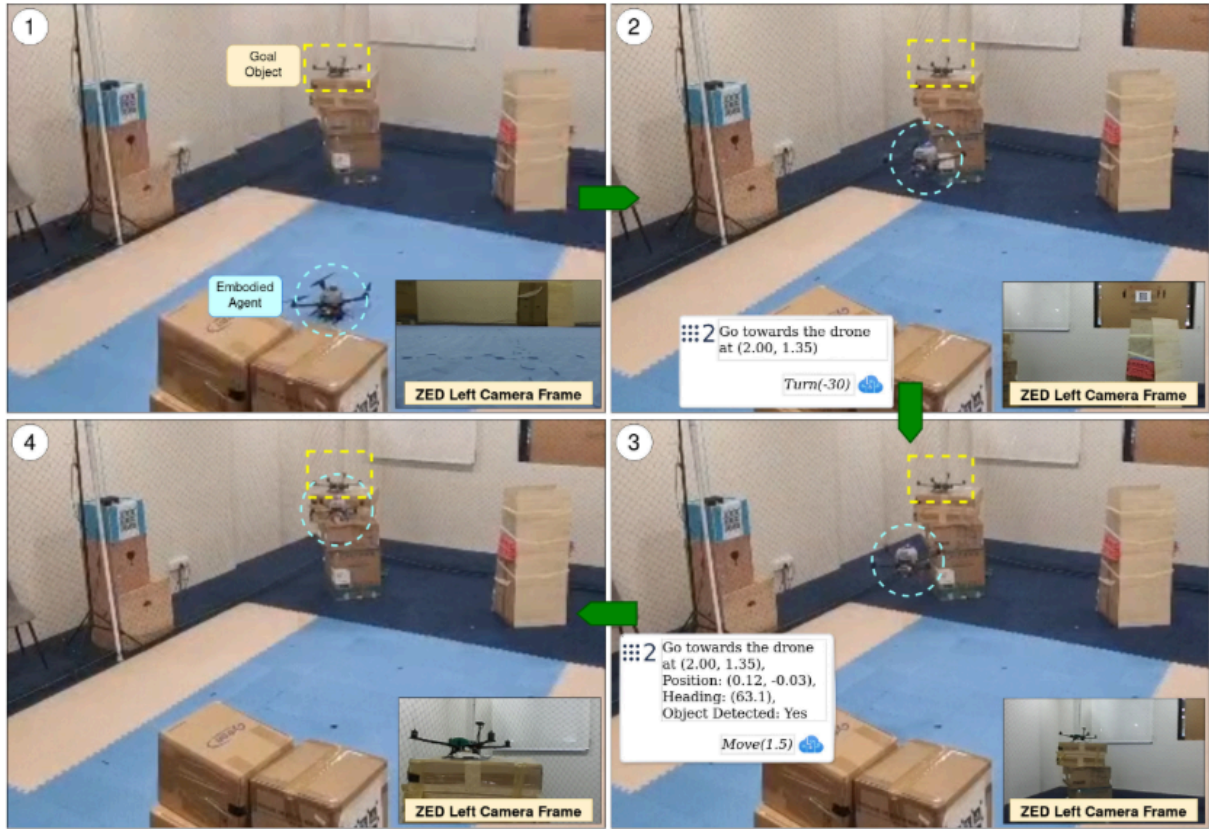
- S_{k+1} : 현재 시점 k 에서 에이전트의 제어 입력 A_k 를 받은 후 다음 이산 시간 단계 $k + 1$ 에서의 쿼드콥터의 새로운 상태를 나타냅니다.
- PX4 : 쿼드콥터의 상태 변화를 결정하는 PX4 자동조종 장치(autopilot) 시스템의 동역학을 나타냅니다. PX4는 드론의 비행 제어 및 안정화를 담당하는 핵심 소프트웨어입니다.
- S_k : 현재 이산 시간 단계 k 에서의 쿼드콥터의 상태를 나타냅니다. 이 상태에는 드론의 위치, 속도, 자세 등이 포함될 수 있습니다.
- A_k : 이산 시간 단계 k 에서 에이전트가 쿼드콥터에 발행하는 제어 입력을 나타냅니다. 이 제어 입력은 드론이 어떻게 움직여야 하는지에 대한 명령(예: 이동 거리, 회전 각도)을 포함합니다.

$$A_k = \mathcal{LLM}(C_k, \mathcal{H}_k^{(N)}, \text{VLM}(\mathcal{V}_k))$$

에이전트 동작 결정 공식

- C_k (Contextual Queries): 시간 단계 k 에서 에이전트에게 제공되는 임무 관련 문맥적 질의를 나타냅니다. 이는 드론의 현재 상태 메타데이터, 주어진 임무 지침, 환경 단서 등을 포함할 수 있습니다. 예를 들어, "드론아, 표적을 찾아 0.5m 반경 안으로 접근해"와 같은 초기 임무 지시나 중간 단계의 지침이 될 수 있습니다.
- $H_k^{(N)}$ (History of Action Commands): 에이전트가 최근에 실행한 N 개의 행동 명령 이력을 나타냅니다. 이 이력은 LLM이 드론의 연속적인 행동 흐름을 이해하고, 이전 명령과의 연관성을 고려하여 다음 행동을 결정하는 데 도움을 줍니다. 본 연구에서는 최근 5개의 유효한 명령 이력이 LLM에 제공되었습니다.
- $VLM(\cdot)$ (Vision Language Model): 시각-언어 모델을 의미합니다. 이 모델은 드론의 온보드 카메라에서 얻은 시각적 관측(V_k)을 평가하여 현재 장면에 임무 관련 객체(예: 다른 로봇 또는 드론)의 존재 여부를 판단합니다. 이 논문에서는 Gemma3, Llama3.2-Vision, Llava1.6과 같은 VLM들을 사용합니다. VLM의 출력은 텍스트 형태의 응답(예: "Yes" 또는 "No")으로 LLM에 제공됩니다.
- V_k (Visual Observations): 시간 단계 k 에서 드론의 온보드 카메라로부터 획득한 현재 시각적 관측값(이미지 데이터)을 나타냅니다.

1. C_k : 현재 임무 맥락과 지시사항.
2. $H_k^{(N)}$: 에이전트가 이전에 수행한 행동의 기록.
3. $VLM(V_k)$: VLM이 현재 카메라 시각 정보를 분석하여 얻은 환경에 대한 이해(예: 특정 객체의 존재 여부).



$$A_k = \text{LLM}(C_k, H(N)_k, \text{VLM}(V_k))$$

- A_k : 시간 k 에 에이전트(드론)가 실행하는 제어 입력 또는 명령 (예: Turn, Move).
- LLM : 대규모 언어 모델로, 사용자의 자연어 명령을 해석하고 드론의 행동을 계획하는 주요 추론 엔진입니다.
- C_k : 임무 목표("Go towards the drone at (2.00, 1.35)")와 같은 임무별 상황 질의 및 상태 메타데이터를 포함하는 문맥적 입력입니다.
- $H(N)_k$: 에이전트가 최근 실행한 N 개의 액션 명령 이력을 나타냅니다. 이 정보는 LLM 이 과거 행동을 고려하여 일관성 있는 다음 명령을 생성하는 데 도움을 줍니다.
- VLM : 시각-언어 모델로, 드론의 온보드 카메라(V_k)에서 얻은 현재 시각적 관찰을 처리합니다. VLM 은 이미지로부터 객체 감지("Object Detected: Yes")와 같은 장면 이해 정보를 생성하여 LLM 에 제공합니다.

$$S[x, y, z] \in \mathcal{B}, \quad t \in [0, K_{\max}]$$

쿼드콥터의 작동 제약 조건과 임무 시간을 정의하는 수식

실험 결과

LLM Model	Parameter Size	LLM Valid Commands (%)	VLM Model	Parameter Size	VLM Valid Detections (%)	Mission Success Rate (%)
Gemma3	4B	100	Gemma3	12B	100	40
			Llama3.2-Vision	11B	100	30
			Llava1.6	7B	98	30
Qwen2.5	3B	100	Gemma3	12B	100	30
			Llama3.2-Vision	11B	100	35
			Llava1.6	7B	97	30
Llama-3.2	3B	100	Gemma3	12B	100	30
			Llama3.2-Vision	11B	100	30
			Llava1.6	7B	98	35
DeepSeek-LLM	7B	38	Gemma3	12B	100	0
			Llama3.2-Vision	11B	100	5
			Llava1.6	7B	98	0

- **LLM Model 및 Parameter Size:**

- 이 섹션은 드론에 명령을 생성하는 데 사용된 LLM의 종류(예: Gemma3, Qwen2.5, Llama-3.2, DeepSeek-LLM)와 각 모델의 파라미터 크기(모델의 복잡성과 성능에 영향을 미치는 요소)를 나타냅니다.
- LLM은 사용자의 자연어 명령을 해석하여 드론이 실행할 수 있는 구체적인 비행 명령(예: Turn, Move)을 생성하는 역할을 합니다.

- **LLM Valid Commands (%):**

- LLM이 생성한 명령 중 문법적으로 올바르고, 미리 정의된 비행 명령 형식 및 파라미터 범위(예: 회전 각도, 이동 거리)를 준수한 명령의 비율을 나타냅니다.
- 이 수치가 높을수록 LLM이 드론 제어 시스템과의 인터페이스를 얼마나 잘 따르는지를 보여줍니다.

- **VLM Model 및 Parameter Size:**

- 이 섹션은 드론이 주변 환경을 시각적으로 이해하고 특정 객체를 감지하는 데 사용된 VLM의 종류와 파라미터 크기를 나타냅니다.
- VLM은 드론의 온보드 카메라로부터 얻은 시각적 관측(visual observations)을 처리하여 임무 관련 객체의 존재 여부를 판단합니다.

- **VLM Valid Detections (%):**

- VLM이 드론의 카메라 시야 내에서 특정 객체(예: 다른 로봇, 드론, 쿼드콥터)의 존재 여부를 "Yes" 또는 "No"로 정확하게 감지한 비율을 나타냅니다.
- 이 수치는 VLM의 시각적 인식 정확도를 의미합니다.

- **Mission Success Rate (%):**

- LLM과 VLM의 협력을 통해 드론이 주어진 임무(예: 목표 객체를 찾아 0.5m 반경 내로 접근)를 성공적으로 완료한 비율을 나타냅니다.
- 임무 실패는 장애물 충돌, 운영 경계 위반, 최대 실행 시간 초과 등으로 정의됩니다.

주요 평가 결과:

- **LLM 성능 비교:** Gemma3, Qwen2.5, Llama-3.2 LLM은 모두 100%의 'LLM Valid Commands' 비율을 기록하여, 시스템이 요구하는 형식에 맞춰 정확하게 명령을 생성하는 능력이 뛰어남을 보여주었습니다. 반면, DeepSeek-LLM은 38%로 현저히 낮은 유효 명령 생성률을 보여, 드론 제어에 사용하기에는 부적합함을 시사합니다.
- **VLM 성능 비교:** Gemma3 (12B), Llama3.2-Vision (11B), Llava1.6 (7B) VLM 모두 97%에서 100%에 이르는 높은 'VLM Valid Detections' 비율을 달성하여, 주어진 객체 감지 임무를 성공적으로 수행했습니다.
- **임무 성공률과 모델 조합:**
 - LLM과 VLM의 조합에 따라 임무 성공률은 상이하게 나타났습니다.
 - 가장 높은 임무 성공률(40%)은 Gemma3 LLM과 Gemma3 VLM 조합에서 관찰되었습니다. 이는 LLM의 유효 명령 생성 능력과 VLM의 정확한 객체 감지 능력이 드론의 임무 수행에 필수적임을 보여줍니다.
 - 특히 DeepSeek-LLM은 낮은 'LLM Valid Commands' 비율(38%)로 인해 Gemma3 VLM과의 조합에서 0%, Llama3.2-Vision VLM과의 조합에서 5%의 매우 낮은 임무 성공률을 기록했습니다. 이는 LLM이 유효한 명령을 생성하는 것이 임무 성공에 얼마나 결정적인지를 명확히 보여줍니다.
- **임무 실패 원인:** 유효 명령 생성 및 객체 감지율이 높았음에도 불구하고, 일부 임무 실패는 LLM이 생성한 명령 값의 비최적화(예: 목표 반경 0.5m를 과도하게 넘거나 미달함) 또는 VLM의 오탐지(false positive/negative)로 인해 발생했습니다. Llava1.6 모델은 유일하게 "Yes", "No" 외의 응답을 생성하는 경우도 있었습니다.

<https://github.com/limshoonkit/ros2-px4-agent-ws>