

자율주행 주차로봇 운영 소프트웨어 성능 테스트 및 최적화

한밭대학교 SW중심대학 산학연계프로젝트
(Eco AI)

참여학생: 박범도, 길기훈, 백민우, 정민성, 권우현, 장하린, 장여진

지도교수: 이상금

참여기업: (주) 아군

Leveraging Superabsorption Gain in Quantum Batteries via
Graph Neural Network-Based Reinforcement Learning

Contents

01 연구 배경

02 방법론

03 요약

04 성과 및 계획

● 문제 정의

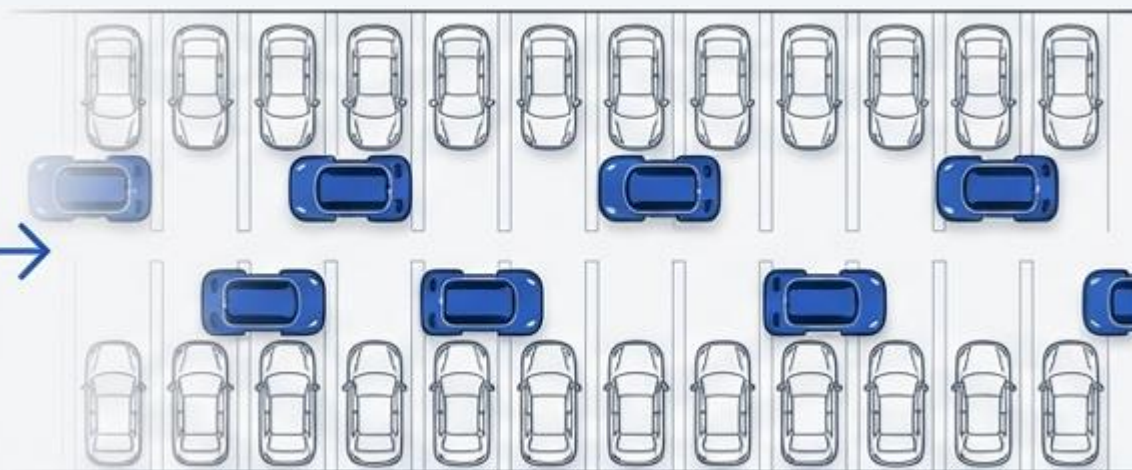
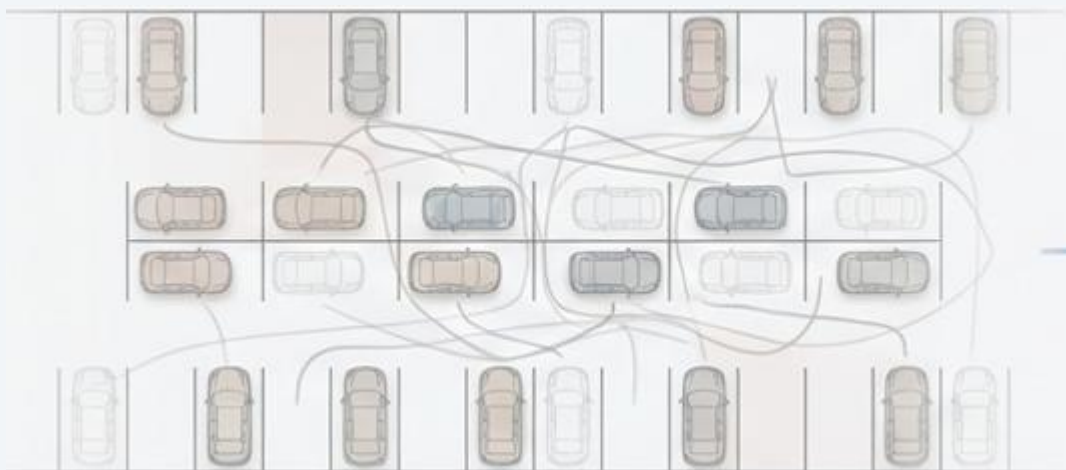
도심 주차장의 비효율 및 위험성

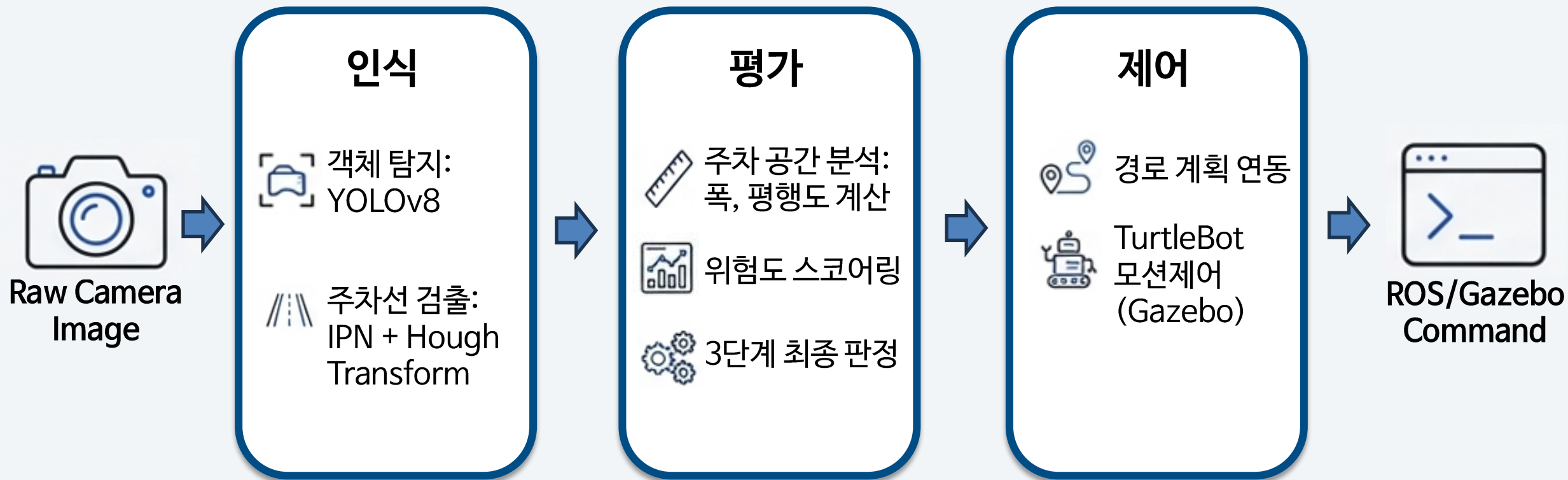
- 공간 비효율: 승하차 공간으로 인한 자주식 주차장 대비 약 30% 공간 낭비
- 안전 문제: 주차 공간 탐색 중 보행자 접촉 사고, 주차 이후 문콕 사고 등의 잠재적 위험 존재

● 프로젝트 목표

자율주행 주차로봇을 활용한 해결방안

- 실제 환경 인지능력: 야간, 우천, 역광 등 실제환경의 제약을 극복하는 자율주행 시스템 개발
- Sim- to-Real 격차 해소: 시뮬레이션에서 검증된 알고리즘이 실제 로봇에서 안정적으로 동작하도록 시스템 최적화





자율주행 주차로봇 시스템 아키텍처

02 방법론

- 객체 인식

기존 YOLOv5 모델: 야간, 우천 역광 등 비정형 환경에서의 인식 성능 저하 문제 발생

- 해결 방안

1. YOLOv8: 기존 YOLOv5 모델 대비 정확도가 약 33% 향상된 모델 도입
2. 데이터 중심 접근: 실제 주행 환경 데이터 수집 및 데이터 증강 적용으로 환경 인식 강인성 강화



TensorBoard 분석 및 K-means 클러스터링을 활용한 Anchor Box 재산출로 모델 최적화.



YOLOv5



YOLOv8

YOLOv5 모델과 YOLOv8 모델 비교 예시

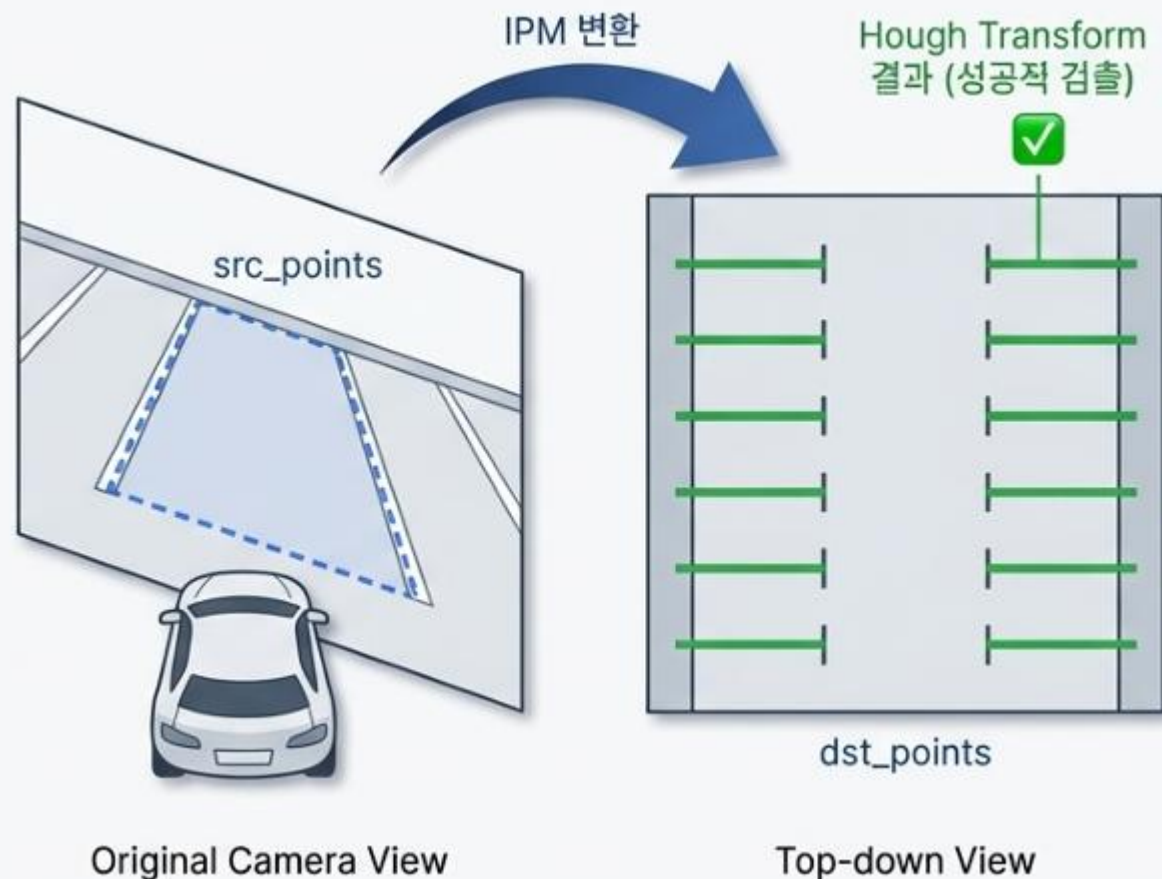
• Inverse Perspective Mapping (IPM)

카메라의 원근 왜곡을 제거하여 이미지를 하늘에서 내려다보는 듯한 Top-down View로 변환

- 모든 주자선을 동일한 스케일에서 분석

```
# CameraConfig dataclass
src_points = np.float32([[250, 450], ...])
dst_points = np.float32([[0, 0], ...])

# Perspective Transform
M_ipm = cv2.getPerspectiveTransform(src_points, dst_points)
```



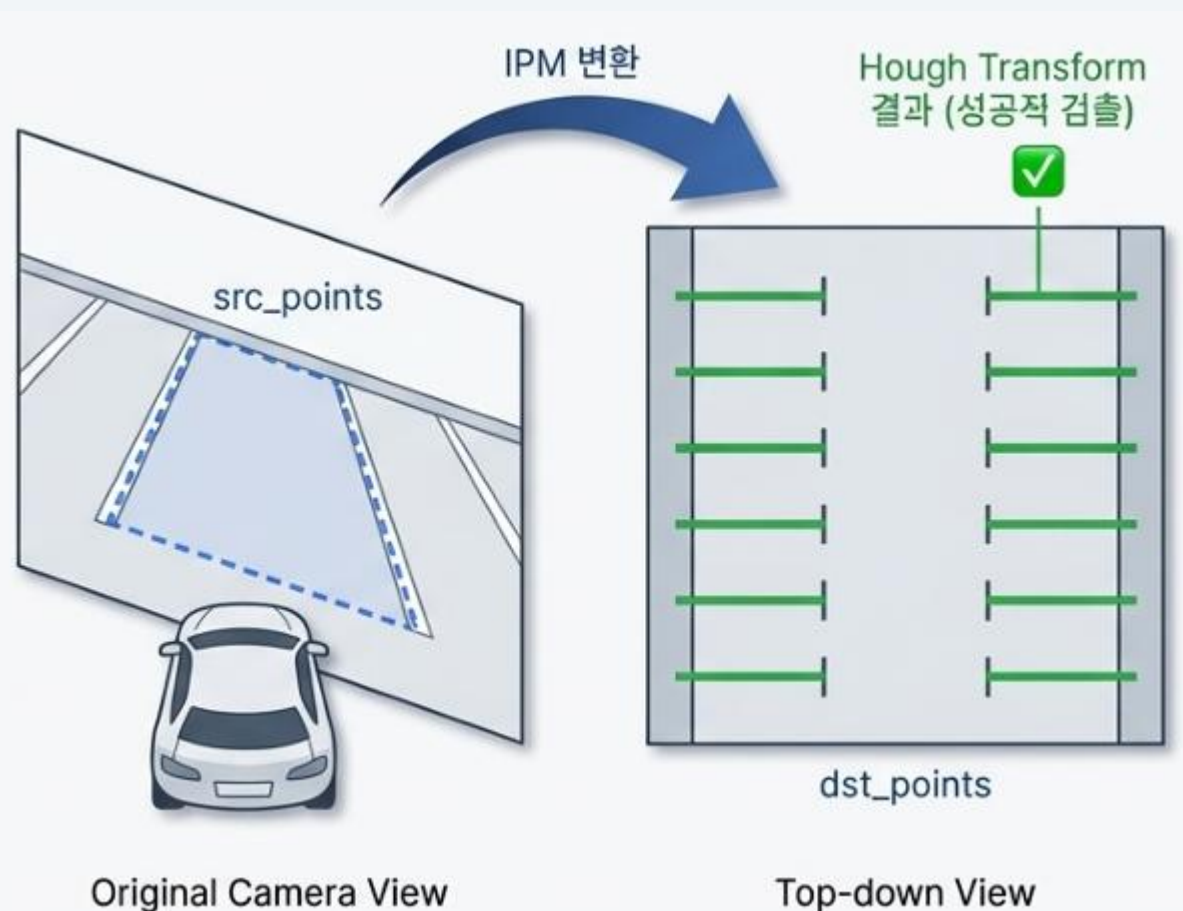
• Optimized Hough Transform

Top-down 이미지에서 직선성분을 효과적으로 추출

- 노이즈를 최소화하고 주차선만 정밀 검출

```
# CameraConfig dataclass
src_points = np.float32([[250, 450], ...])
dst_points = np.float32([[0, 0], ...])

# Perspective Transform
M_ipm = cv2.getPerspectiveTransform(src_points, dst_points)
```



• 단계별 평가 도입

추출된 데이터를 종합하여 단계적으로 주차 가능한 공간 판정



Stage 1: 주차선 쌍 매칭 (Line Pairing)

기울기가 유사한 두 직선을 하나의 주차공간 후보로 결합



Stage 2: 기하학적 분석 (Geometric Analysis)

- 두 직선 간 거리가 유효범위 내 있는지 확인
- 두 직선의 각도 차이가 허용 오차 이내인지 확인



Stage 3: 위험도 산출 (Risk Scoring)

주차 공간 중앙과 감지된 장애물(차량, 사람) 간의 거리 계산



Stage 4: 최종 판정 (Final Decision)

산출된 위험도 기반으로 3단계 판정 제시

• 3단계 판정 시스템

위험도에 따라 세 가지의 명확한 상태 판정

‘POSSIBLE’ (가능)



Risk Score: < 0.4

주차 공간의 폭과 평행도가 기준을 만족하며, 주변에 위협적인 장애물이 없습니다.
즉시 주차를 시도할 수 있습니다.

‘HOLD’ (보류)



Risk Score: 0.4 ~ 0.7

주차 공간은 유효하나, 장애물이 근접해 있어 잠재적 위험이 존재합니다. 잠시 정지 후 재탐색이 필요합니다.

‘IMPOSSIBLE’ (불가)



Risk Score: > 0.7

주차 공간의 규격이 부적합하거나, 장애물이 경로를 막고 있어 주차가 불가능합니다.


Code as Evidence

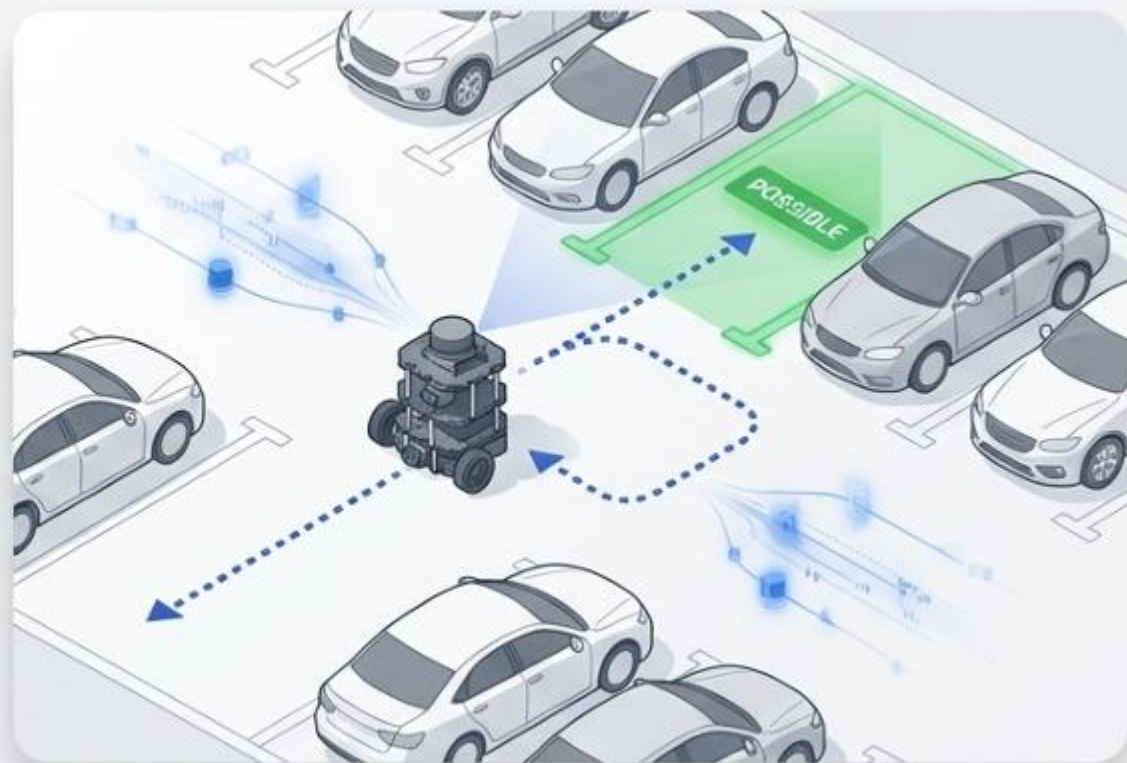
```
class ParkingDecision(IntEnum):  
    POSSIBLE = 0    # GREEN  
    HOLD = 1       # YELLOW  
    IMPOSSIBLE = 2 # RED
```

● 시뮬레이터를 사용한 검증

로봇 운영체제(ROS)와 통합하여 가상 환경(Gazebo)에서 End-to-End 테스트 수행

System Integration Details

-  환경: Ubuntu 기반 ROS1 및 Gazebo 시뮬레이터
-  로봇 모델: TurtleBot3
-  데이터 흐름: Gazebo의 가상 카메라(Image)와 LIDAR 센서 데이터를 ROS Topic(`_camera_callback`)으로 수신하고, 최종 제어 명령을 `cmd_vel` Topic으로 발행하여 로봇을 움직입니다.
- ✓ 검증 내용: 평행 및 수직 주차를 위한 경로 계획 알고리즘(전진, 회전, 후진 순서)의 동작을 시각적으로 검증했습니다.



- 주요 성과 (Project Outcomes)

- 1 고성능 인식 모델 확보

다양한 실제 환경 데이터로 학습된 YOLOv8 기반 객체 검출 모델을 개발하여 환경 변화에 강인한 성능 달성

- 2 강건한 의사결정 로직 구현

기하학적 분석과 위험도 평가를 결합한 3단계 판정을 통해 안정적인 주차 가능 공간 판단 로직 구현

- 3 시뮬레이션 기반 통합 검증

개별 모듈을 ROS/Gazebo 환경에 통합하여 End-to-End 시나리오의 실행 가능성을 성공적으로 검증

- 향후 계획 (Key Learnings)

**“데이터의 품질이
시스템의 성능을 결정한다”**



프로젝트 성공의 가장 중요한 요소는 야간, 우천 등 비정형 환경을 포함한 고품질 데이터의 확보 및 처리였습니다.

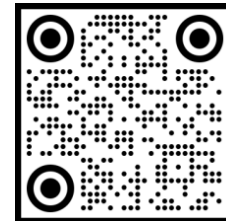
**“Sim-to-Real Gap은
여전히 중요한 과제다”**



시뮬레이션에서의 성공이 실제 환경에서의 성공을 보장하지 않습니다. 센서 오차, 통신 지연 등 현실의 변수를 고려한 추가적인 최적화가 필요합니다.

감사합니다

GitHub Repository



○ 2025.11.30 ○

참여학생: 박범도, 길기훈, 백민우, 정민성, 권우현, 장하린, 장여진 / 지도교수: 이상금 / 참여기업: (주) 아군