청년 AI · BIG DATA 아카데미 25기

크라우드소싱 배송을 위한 배송지 고려 적재 최적화

B반 2조 | 김아연 김한탁 류지연 서지윤 오석훈 하지민

2024 .04. 19

Contents

01

추진 배경

POINT 01

중국 e-커머스의 공습

- 중국 이커머스 플랫폼(C커머스)
  - 알리익스프레스(알리)·테무·쉐인 등
- 쿠팡, 3조원 이상을 투자해 2027년까지 로켓배송(당일·익일배송) 지역 전국 확장 계획

POINT 02

국내 e-커머스의 라스트마일 배송 경쟁

- 국내 e-커머스는 새로운 배송형태인 신 라스트마일 배송 시스템 구축을 계획
- 2016년, 익일배송의 비율이 2%였으나 2022년에는 일반적인 배송 방식으로 자리 잡음

*라스트 마일
: 물류 센터에서 구매자 집 앞까지 주문한 물품이 배송되는 전과정

POINT 03

쿠팡플렉스를 시작으로 일반인 자차 배송 확산

- 일반인 인력을 활용한 자차 배송 시스템이 e커머스 시장 전반으로 확산
- 자차 배송 시스템은 급증하는 주문량에도 쿠팡이 로켓배송 속도를 유지하는 비결의 하나

02

프로젝트 목표 및

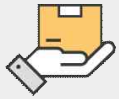
적재 공간 및 배송 경로최적화

강화학습 기반 AI 모델 개발

Unity 시뮬레이션 환경 구축

크라우드소싱 배송을 위한 배송지 고려 적재 최적화

C커머스의 등장으로 국내 e-커머스 기업들의
라스트마일 배송 경쟁이 심화되어
새로운 라스트마일 배송 시스템인 자차 배송
대상으로 적재 및 경로 최적화 필요성 증가



배송 효율성 및 생산성 향상

신규 라이더의 진입장벽 완화

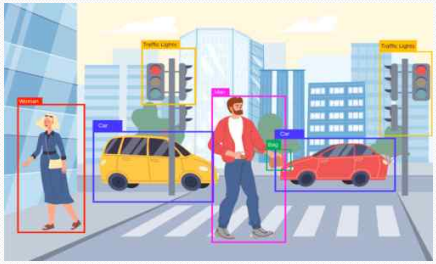
고객만족도 향상

03

활용 기술

POINT 01

Object Detection



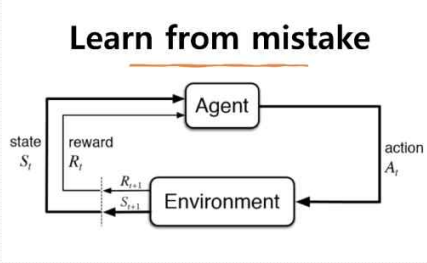
찾고자 하는 객체의 특징을 사전에 추출하고
주어진 영상 내 해당 객체의 특징을 검출하여
식별 및 분류하는 방법

⇓

여러 차량의 이미지를 딥러닝 모델을 이용해 학습
차량의 종류를 인식하여
해당 차량의 트렁크 내 적재 가능 용량을 알려주도록 함

POINT 02

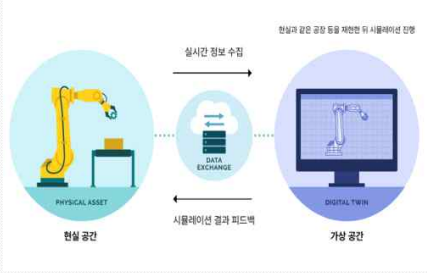
강화학습



- Agent : 로봇팔
- Action : 로봇팔이 상자를 들어 특정 위치로 이동
- Environment : 가상환경 속 물류센터
- State : 자동차 별 트렁크 점유율
- Reward : 쌓은 상자의 부피(무게, 부피가 높을 수록 아래 깔리도록), 배송지의 우선순위가 높은 경우(맨 위에 쌓이도록)

POINT 03

디지털 트윈



컴퓨터에 현실 속 사물의 쌍둥이를 만들고,
현실에서 발생할 수 있는 상황을
컴퓨터로 시뮬레이션함으로써 결과를 미리 예측하는 기술

⇓

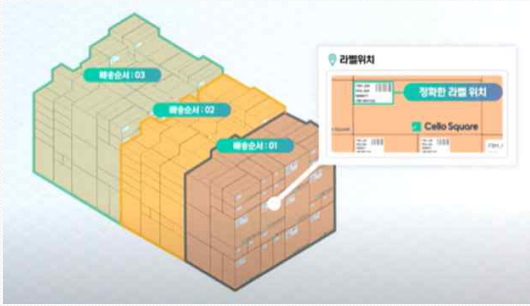
유니티 내 로봇 팔(agent) 강화학습을 통해
차량 내부 부피 공간의 배송지 우선순위를 고려한
최적 적재 알고리즘 적용 및 디지털 트윈 구축

04

기업 사례

POINT 01

삼성 SDS 첼로스퀘어 - 적재 최적화 서비스



- **창고 운영의 생산성과 효율성 향상**
  - 화물을 실을 때 고려해야하는 적재 공간, 배송 순서, 라벨 위치 등 모든 요소를 고려하여 시뮬레이션하고, 화물에 맞는 컨테이너 수량 및 적재율을 안내

POINT 02

LG CNS & UNITY

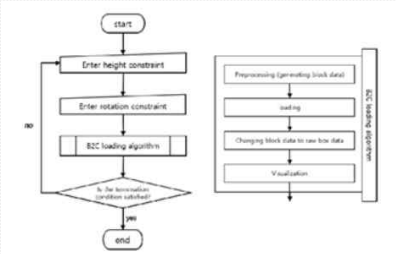
- LG CNS는 유니티의 3D엔진 기반으로 고객사 제조공장 공간과 설비를 가상화
- 가상화 공간에 각종 공장과 물류센터 데이터를 연계해 실제 공장을 원격 운영하는 메타버스 환경을 구축

05

논문 분석

POINT 01

B2B, B2C 물류 환경을 고려한 3차원 차량 적재 알고리즘 개발



- Clustering: 배송 경로 정보와 물건의 배송지 정보를 통해 물건의 배송 권역 구분 및 배송 순서 할당
- Grouping 과정을 통해 배송 권역이 구분된 제약 조건 만족하는 블록 생성
- 정렬된 블록의 순서에 따라 배치를 시작하여 트럭을 모두 채울 때까지 반복

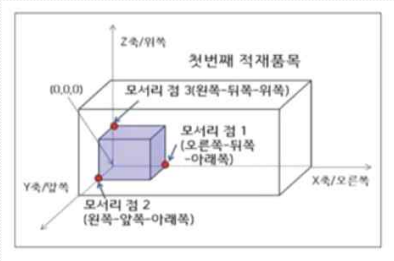
⇓

서로 다른 타입의 적재물 간의 안정적 적재 가능
최적 배송을 위한 적합한 의사결정 가능

유명재, 이태환, 송병욱, 이상덕. (2022). B2B, B2C 물류 환경을 고려한 3차원 차량 적재 알고리즘 개발. 로지스틱스연구, 30(6), 89-103.

POINT 02

배송 지점과 품목의 우선순위 제약 조건하 3차원 품목 적재 해법



- 배송 지점의 우선순위와 도착순서, 품목의 우선순위를 고려
- 타부 서치를 적용하여 추가 품목의 적재 가능 여부를 확인 후 추가 적재가 가능한 적재 방법을 선택하는 휴리스틱 방법론을 사용한 적재 방안

⇓

지점 방문 및 품목 우선순위 제한 사항을 포함한 제시를 통해
다양한 문제 적용 가능

*타부서치: 한정된 시간 내에 최적의 해 대신 현실적으로 만족할 만한 수준의 해를 구함
*휴리스틱 알고리즘: 경험, 규칙, 또는 추정치와 같은 지식을 기반으로 문제 해결을 위한 가이드 라인을 제시하여 어려운 문제 해결

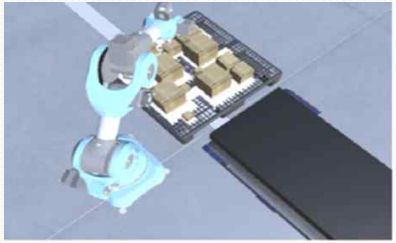
송수빈, 이동길. (2014). 배송 지점과 품목의 우선순위 제약 조건하 3차원 품목 적재 해법. 로지스틱스연구, 22(1), 17-28. 10.15795/kjs.2014.22.1.002

06

전기수 분석

POINT 01

21기 C3조 PBA Twins



- 강화학습 기반 디지털 트윈을 활용한 물류 최적화 시스템
  - 주요기능: Unity(가상환경 구현), PPO(강화학습 알고리즘), Bin Packing

POINT 02

22기 C4조 'CPost(C4st)'

- 강화학습을 통한 3D 물류 적재 최적화 시스템
  - 주요기능: Unity(가상환경 구현), ACTKR(강화학습 알고리즘), ROS(로봇팔 제어), 3D Bin Packing System(3차원 물류 적재 시스템)

차별점

- 밀도와 무게만을 고려하여 최적 적재 방안을 도출 -> 밀도, 무게 뿐만 아니라 배송지의 우선순위를 고려하여 물건의 최적 적재 방안을 도출
- 표준 산업용 팔레트 사이즈인 110cm x 80cm x 170cm를 기준 -> 차 종에 따른 트렁크의 실제 크기를 바탕으로 다양한 팔레트 크기를 새롭게 설정

07

시나리오

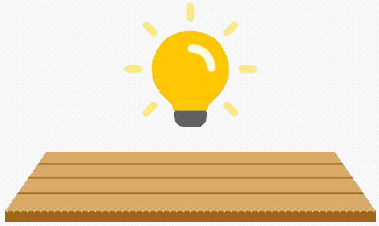
Unity

POINT 01

차량 Object Detection -> 차종 인식

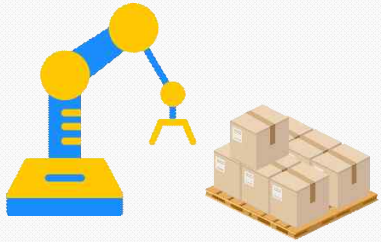
POINT 02

차종에 맞는 트렁크 부피 공간 생성



POINT 03

Random 박스 사이즈에 대한
강화학습 알고리즘 기반
적재 최적화 및 배송 경로 최적화



진행 상황

Object Detection

	1-3주차 진행 상황
✓	AIHUB '자동차 차종/연식/번호판 인식용 영상' 데이터 수집
✓	차량 종류별 사진 라벨링(Roboflow 이용)
✓	위를 활용한 Yolov8 모델 학습(epoch=200)
✓	모델 성능 확인

	4주차 진행 상황
✓	차종에 따른 트렁크 부피 값 반환할 수 있도록 함
✓	최종 모델 성능 개선(바운딩 박스, 정확도 등)

진행 상황

Algorithm

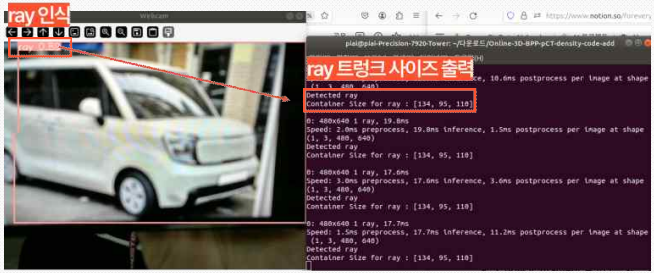
	1-3주차 진행 상황
✓	전 기수(22기 C4조, 23기 B3조), 논문 원본 제약조건 수정
✓	알고리즘 3가지(a3c, policy_gradient, TRPO) 추가
✓	box 랜덤 난수 조건 추가
✓	continous domain 에서의 밀도 제약 조건 추가

	4주차 진행 상황
✓	Object Detection 모델과의 연동을 통한 트렁크 사이즈별 팔레트 구현
✓	배송지 제약조건 추가한 Reward 함수 수정
✓	배송지, 밀도 제약조건 추가 후 3D 시각화
✓	에피소드 중 Reward가 가장 높은 에피소드 선정 후 Best, Worst 3D 시각화

진행 상황

Algorithm

4주차 진행 상황 ✓ Object Detection 모델과의 연동을 통한 트렁크 사이즈별 팔레트 구현




```python
# YOLO 모델을 불러옴
model = YOLO('/home/pial/다운로드/Online-3D-BPP-PCT-density-code-add/best.pt')
cap = cv2.VideoCapture(0)

# 감지된 자동차에 따른 컨테이너 크기 저장
car_container_sizes = {
    'carnival': [123, 231, 100],
    'ray': [134, 95, 110],
    'soul': [185, 126, 83],
    'sportage': [183, 85, 73],
    'stares': [162, 237, 134],
    'tucson': [193, 159, 77]
}

# 실시간 비디오 스트림 처리
while cap.isOpened():
    ret, frame = cap.read()
    if ret:
        results = model(frame)
        cv2.imshow("Webcam", results[0].plot())

        if len(results) > 0:
            filtered_results = filter_boxes(results[0].boxes, results[0].names, min_confidence=0.5)
            annotated_frame = frame.copy()
            for name, conf, bbox in filtered_results:
                bbox = [int(coord) for coord in bbox]
                cv2.rectangle(annotated_frame, (bbox[0], bbox[1]), (bbox[2], bbox[3]), (0, 255, 0), 2)
                cv2.putText(annotated_frame, f'{name} {conf:.2f}', (bbox[0], bbox[1] - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 255, 0), 2)

        for result in results:
            clist = result.boxes.cls
            cls = set()
            for cno in clist:
                cls.add(model.names[int(cno)])

            for car in cls:
                if car in car_container_sizes:
                    print(f"Detected {car}")
                    container_size = car_container_sizes[car]
                    print("Container Size for", car, ":", container_size)
                    # cv2.destroyAllWindows()  # 모든 OpenCV 창을 닫음
                    # cap.release()  # 카메라 자원 해제
                    break  # 반복문 종료

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
```

진행 상황

Algorithm

4주차 진행 상황 ✓ 배송지 제약조건 추가한 Reward 함수 수정

- Reward 함수 수정 내용

```
# 배송지 번호에 따른 이상적인 z 위치 계산 (예시로, 더 낮은 번호가 더 높은 위치에 배치되어야 한다고 가정)
def calculate_destination_reward(self, box, max_height):
    ideal_z = (10 - box.destination_id + 1) * (max_height / 10)  # 10은 배송지 ID의 최대값
    # 실제 z 위치와 이상적인 z 위치의 차이를 기반으로 보상 계산
    reward = max(0, 1 - abs(ideal_z - (box.z + box.lz)) / max_height)
    return reward
```

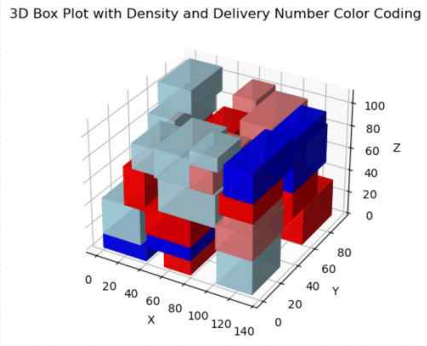
```
reward = box_ratio * 10 + 0.01 * box_density * (self.bin_size[2] - z)
destination_reward = self.calculate_destination_reward(packed_box, self.space.height)
reward += destination_reward  # 최종 보상에 배송지 번호 보상 추가

done = False
info = dict()
info['counter'] = len(self.space.boxes)
return self.cur_observation(), reward, done, info
```

진행 상황

Algorithm

4주차 진행 상황 ✔ 배송지, 밀도 제약조건 추가 후 3D 시각화

- 추가한 밀도, 배송지 우선순위 제약조건이 잘 적용되었다는 것을 확인할 수 있음



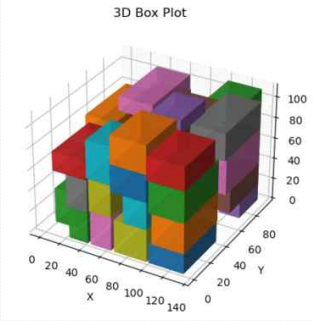
- 연한 빨간색: 밀도가 높고(0.6 초과) 배송지 번호 낮은 것(7 초과)
- 빨간 색: 밀도가 높고(0.6 초과) 배송지 번호 높은 것(7 초과)
- 파란색: 밀도가 낮고(0.6 이하) 배송지 번호 낮은 것(7 이하)
- 연한 파란색: 밀도가 높고(0.6 이하) 배송지 번호 높은 것(7 초과)

진행 상황

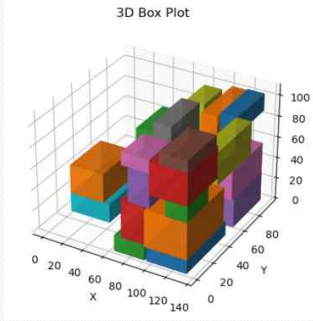
Algorithm

4주차 진행 상황 ✓ 에피소드 중 Reward가 가장 높은 에피소드 선정 후 Best, Worst 3D 시각화

- 위의 Reward 함수를 통해 적재 잘 되었다는 것을 확인할 수 있음



- Best Reward(0.64894) 경우 상자 적재 3D 시각화 ⇒ 박스 37개

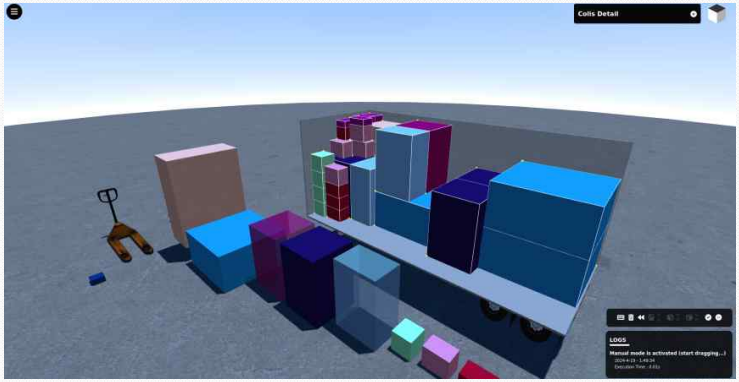


- Worst Reward(0.37074) 경우 상자 적재 3D 시각화 ⇒ 박스 26개

진행 상황

Algorithm

4주차 진행 상황 ✓ javaScript, js를 통해 온라인 환경에서 3D Bin Packing 시뮬레이션



진행 상황

Unity

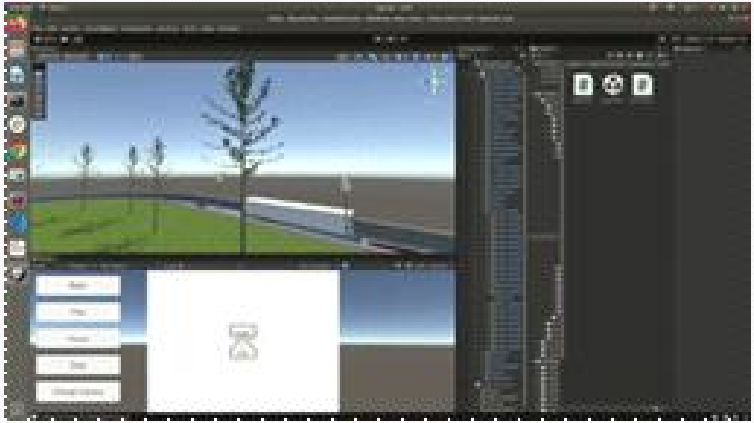
	1-3주차진행 상황
✓	Unity 외 다른 디지털 트윈 환경 분석
✓	유니티 환경 Setting & Unity와 ROS 통신
✓	전 기수(22기C4조) 시뮬레이션
✓	유니티 내 Yolov5 모델 구현

	4주차 진행 상황
✓	Yolov5 모델에 차종 객체 인식 C#스크립트 작성
✓	강화학습 알고리즘 연동 로봇팔 시뮬레이션
	차량 인식 후 차종에 맞는 트렁크 공간 생성
	물류공장 POSCO 맵, 실습실 맵 제작

진행 상황

Unity

4주차 진행 상황 ✓ 유니티 내 Object Detection 연동

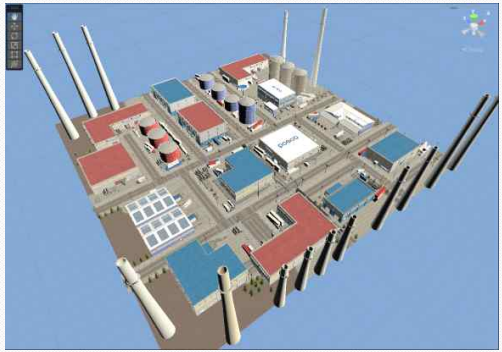


진행 상황

Unity

4주차 진행 상황 ✔ 맵 제작 상황

- 물류공장 POSCO 맵



- VR - 실습실 맵

이상으로 발표를 마치겠습니다. 감사합니다!

THANK YOU FOR ATTENTION

2024. 04. 12