기융프 진행 보고

School of Mechanical and Control Engineering

Prof. Y.K. Kim

22200757 Joungbin Choi

2025.10.22

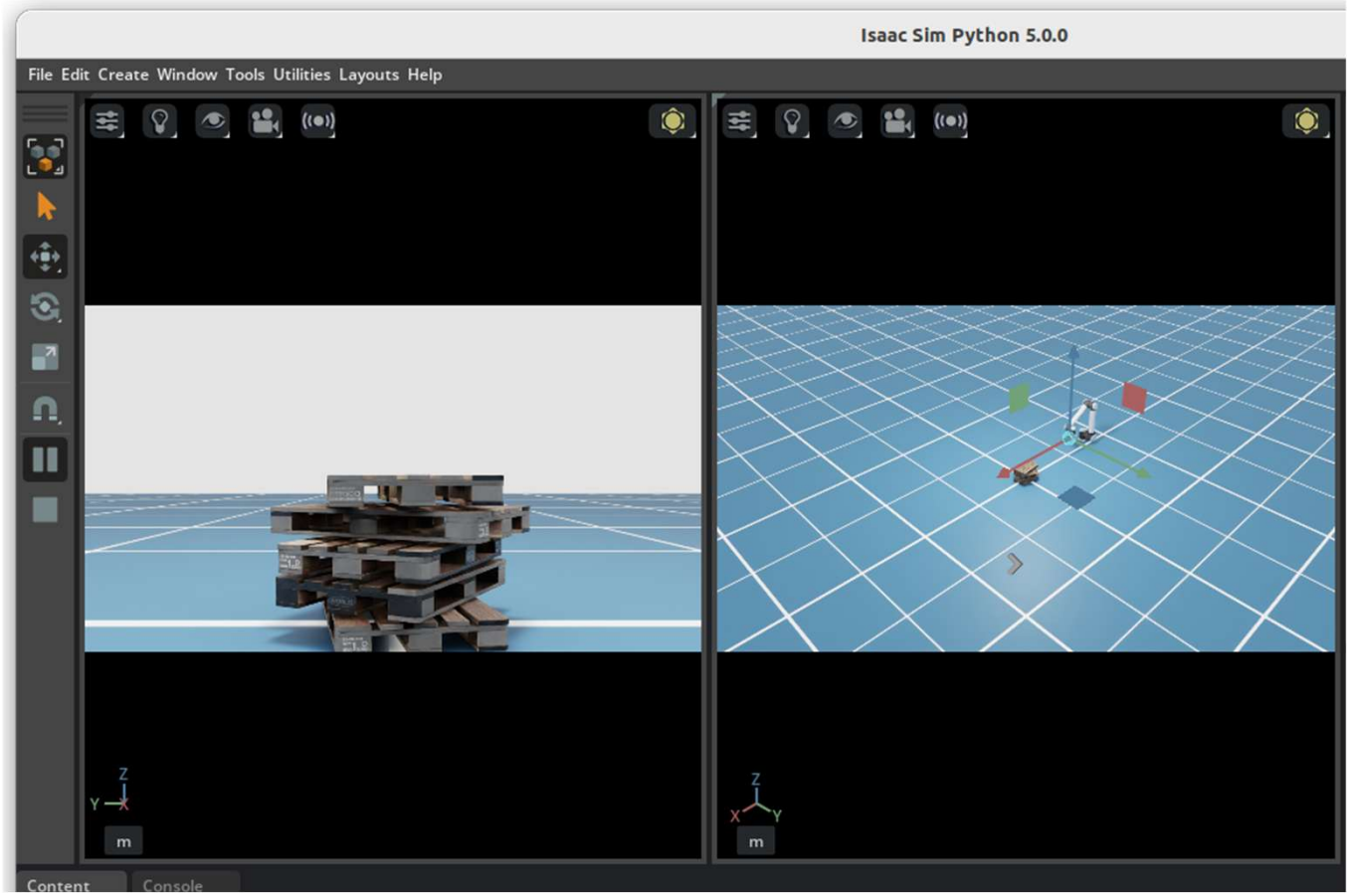
목차

1. 시뮬레이션 환경 구축

Nvdia사의 시뮬레이션 툴인 Isaac Sim 사용

Isaac Sim의 장점

1. 고정밀 센서 시뮬레이션
    1. RGB/Depth/LiDAR/IMU 시뮬레이션
    2. 렌즈 왜곡, 노이즈, FoV, FPS 조절 가능
2. 디지털 트윈 구축
    1. 실제 환경을 디지털로 재현
    2. 환경 재현 가능
3. Isaac Orbit 통합
    1. 강화학습과 Sim을 연결해주는 프레임워크



1. 시뮬레이션 환경 구축

미션 : 산업 현장에서 랜덤하게 쌓여 있는 나무 팔렛을 인식하여 자율적으로 다시 정렬되게 쌓기

시뮬레이션 구성 리스트:

- **나무 팔렛 5개 (Random Pose)**
- **UR10 (Robot Arm)**
- **Depth Camera (for image process and point cloud)**
- **Dome 조명**

2. 모델 데이터와 학습

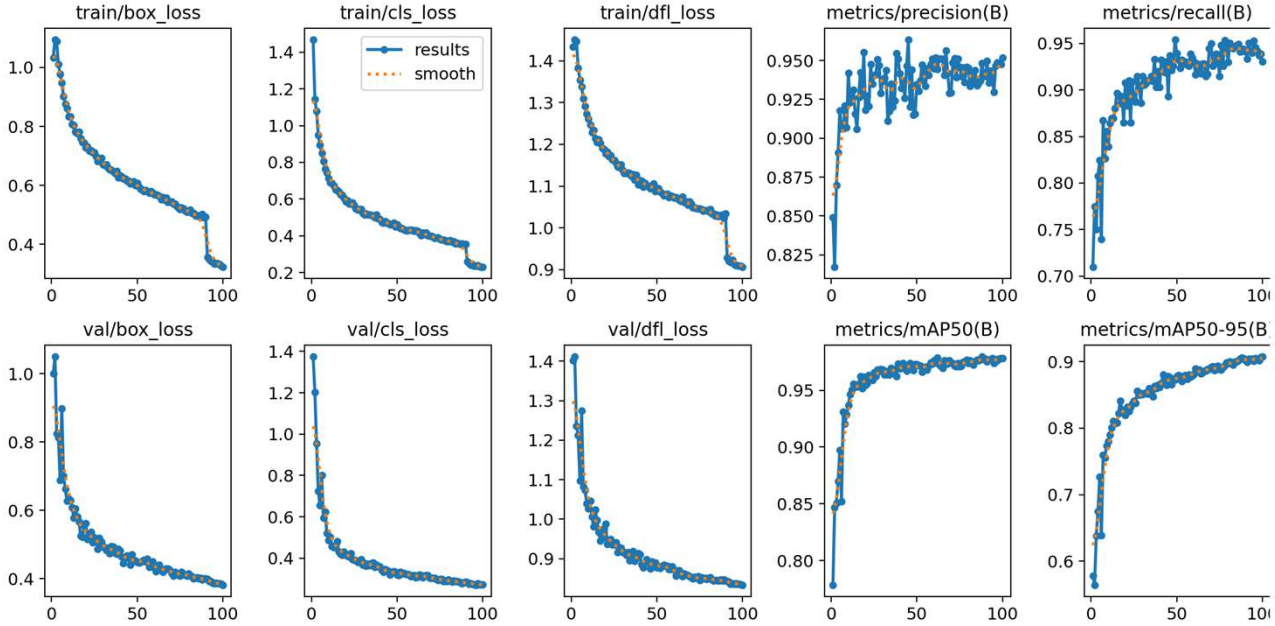
Roboflow에 있는 팔렛 데이터 사용

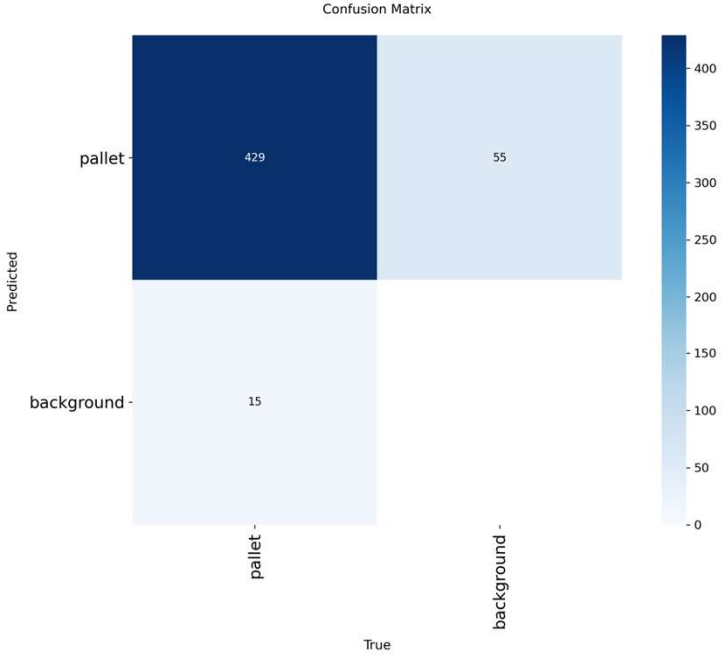
학습 데이터 : https://universe.roboflow.com/esprit-moito/mask_rcnn-73vt5

모델 : Yolov8n

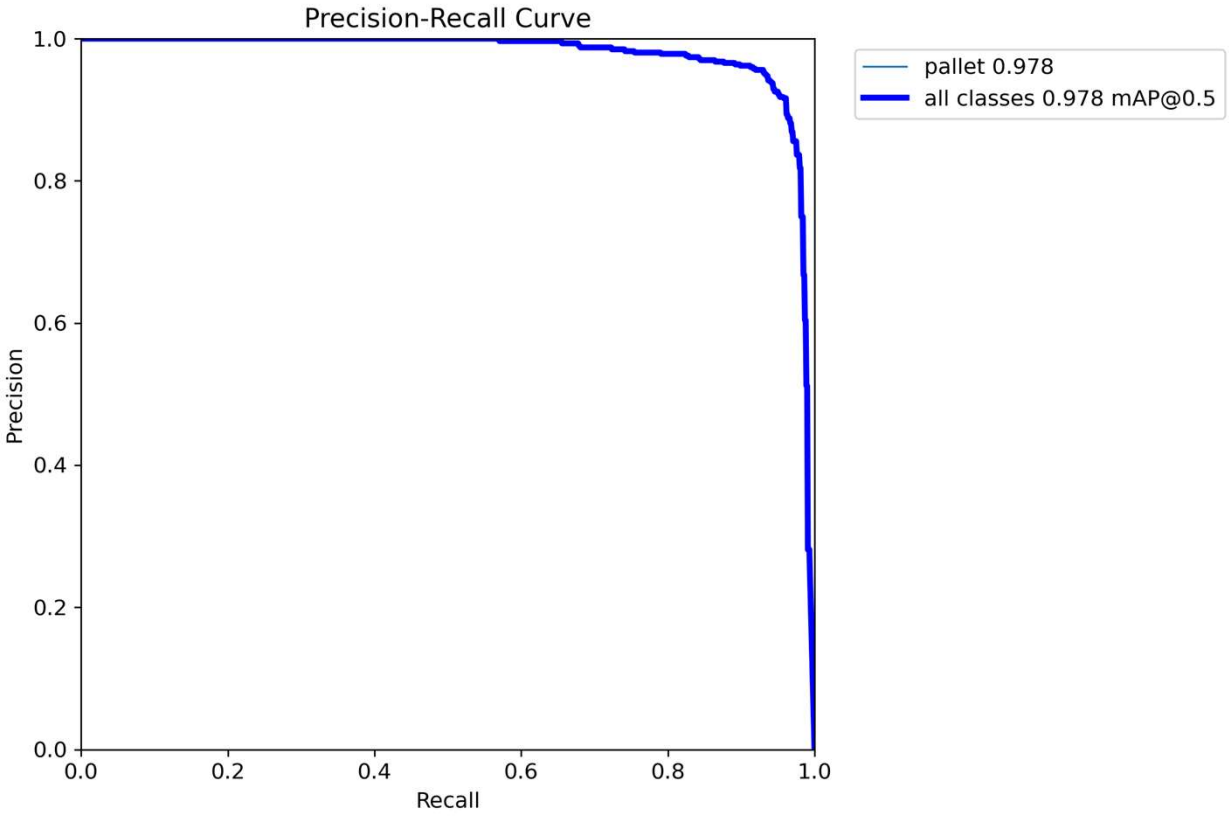
```
results = model.train(
    data='C:/isaacsim/output_data/data.yaml',
    epochs=100,
    imgsz=640,
    batch=4,
    name='pallet_run_1014',
    patience=20,
    workers=0
)
```

2. 모델 데이터와 학습


train/box_loss
train/cls_loss
results
smooth
train/dfl_loss
metrics/precision(B)
metrics/recall(B)
val/box_loss
val/cls_loss
val/dfl_loss
metrics/mAP50(B)
metrics/mAP50-95(B)

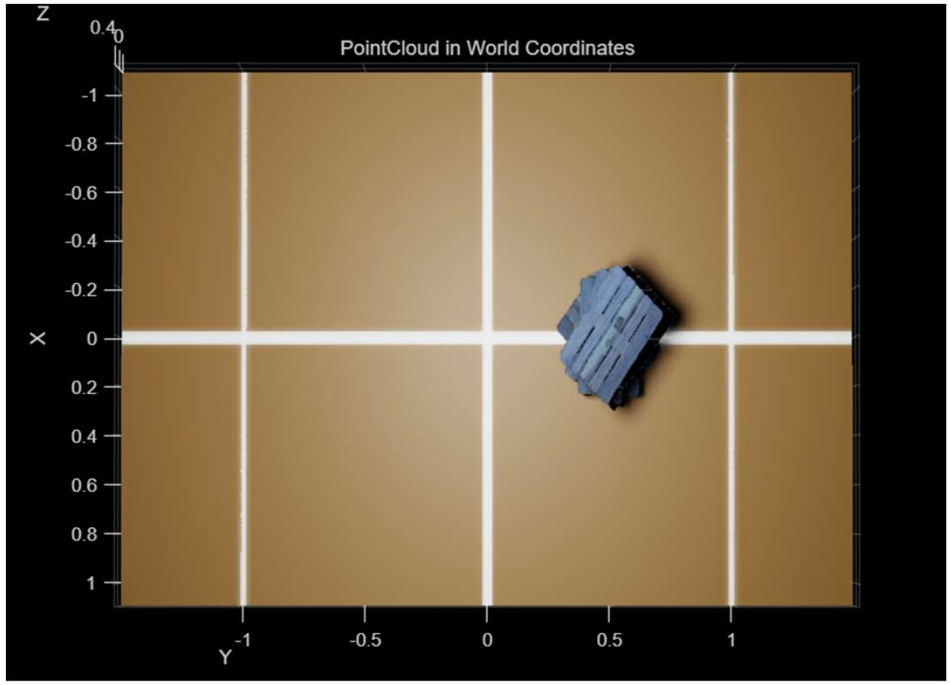

Confusion Matrix
pallet
background
429
55
15
Predicted
True
pallet
background

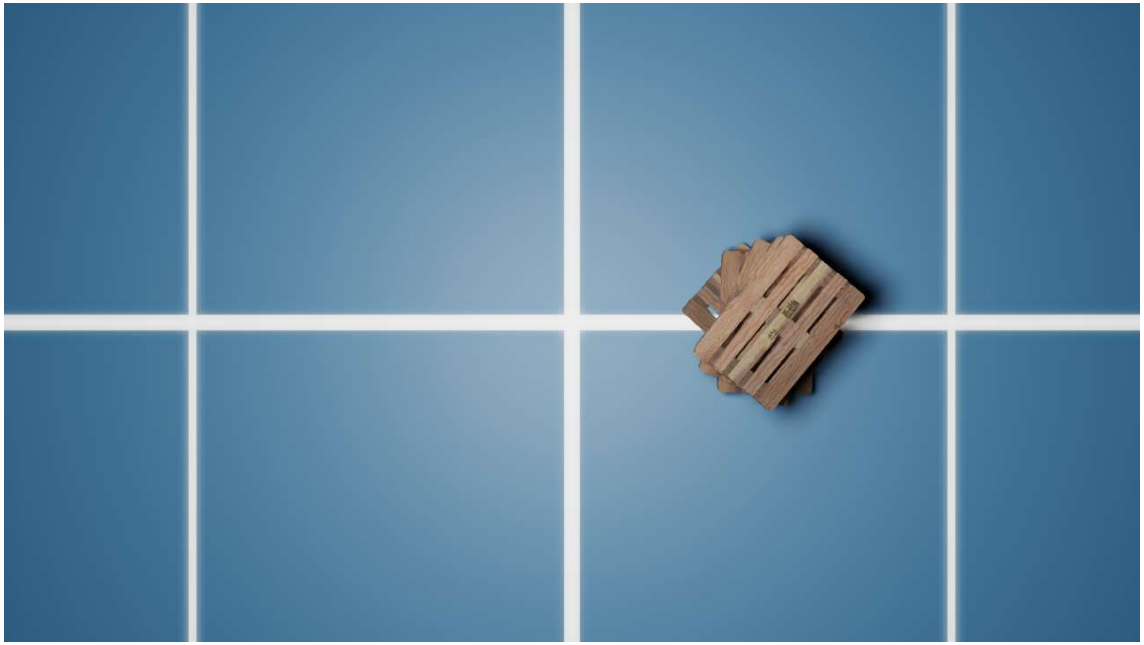
2. 모델 데이터와 학습


Precision-Recall Curve
pallet 0.978
all classes 0.978 mAP@0.5
Precision
Recall
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0


pallet 0.9
pallet 1.0
pallet 0.8

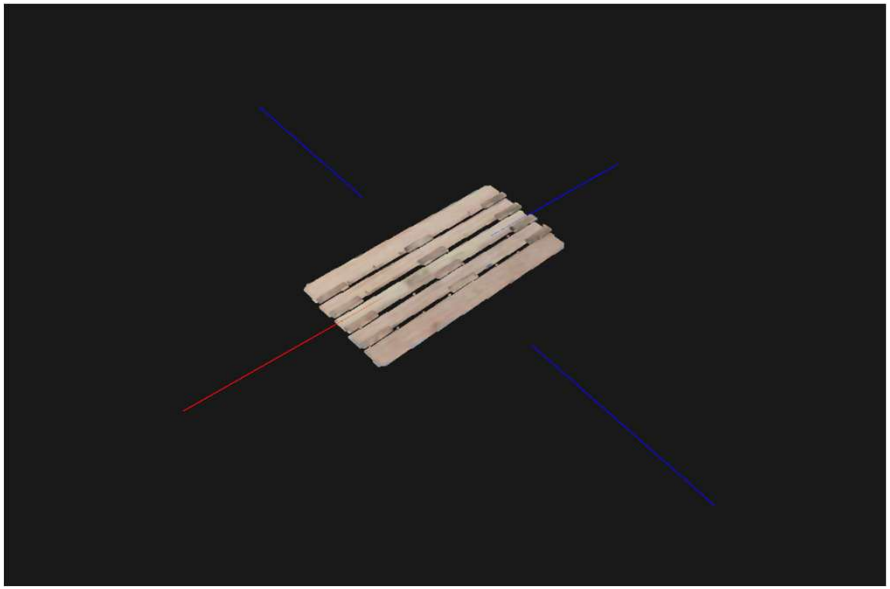
3. 뎁스 카메라와 포인트 클라우드 처리

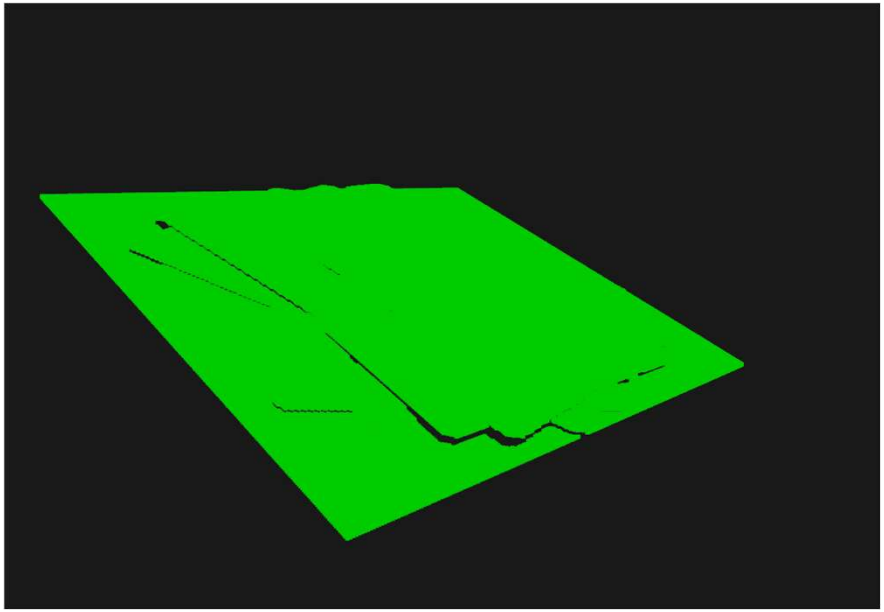




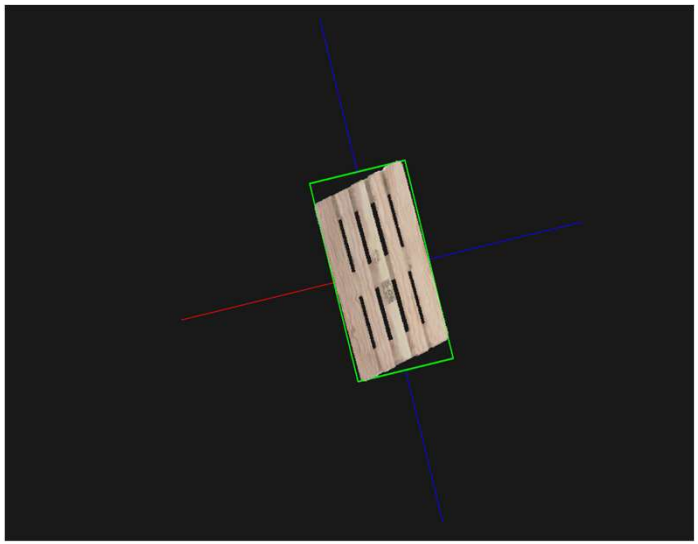
3. 뎁스 카메라와 포인트 클라우드 처리

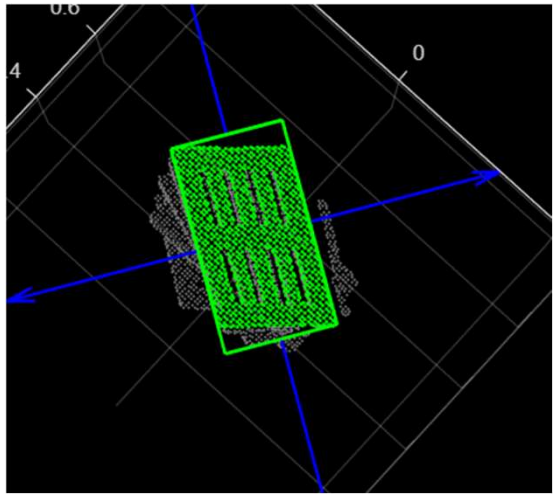
$$Cam\ to\ world\ matrix = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 5 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$





3. 뎁스 카메라와 포인트 클라우드 처리





OBB기반 자세 추정

OBB(Oriented Bounding Box): AABB는 무조건 axis에 평행한 상자를 생성하는 반면 OBB는 기울어진 상자에 맞춰 상자를 생성

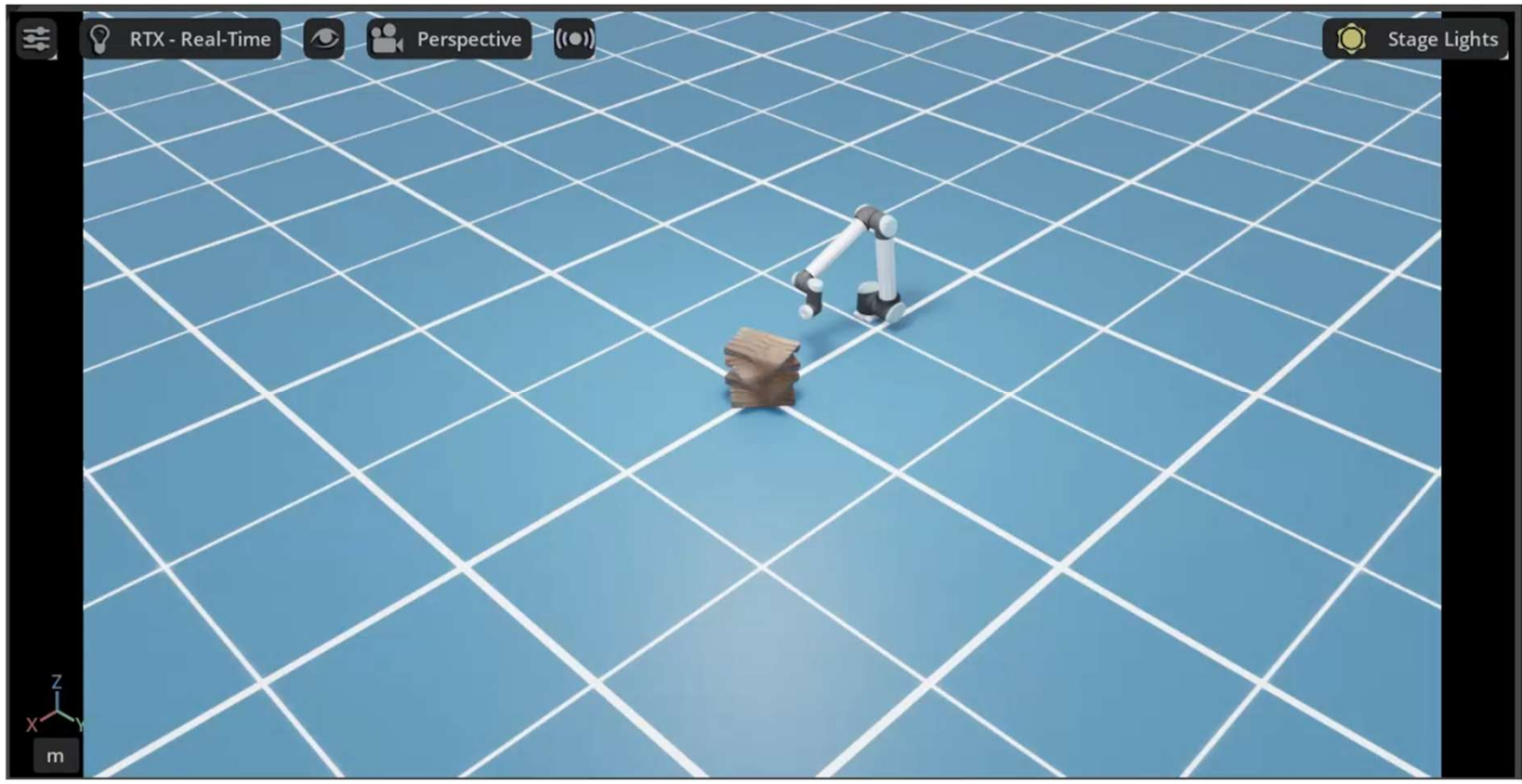
SOR을 통해 아웃라이어 제거(필터링)

PCA(주성분 분석) 기반으로 포인트들의 무게 중심을 구하고 공분산 행렬을 통해 고유벡터를 구한다.

구한 상자 사이드 면의 법선 벡터를 구한다.

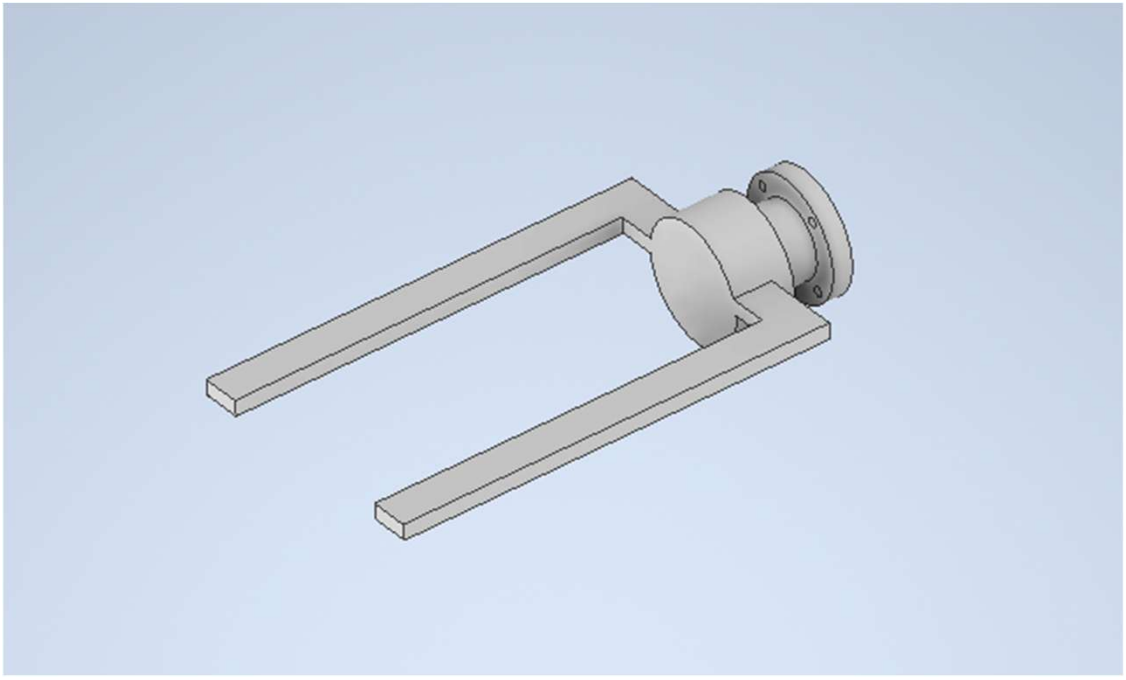
4. 로봇팔 로직

1. 로봇과 가장 가까운 옆면을 찾는다.
2. 최종 목표 위치와 자세를 설정한다.
3. 이후 x축으로 offset을 주어 사전 접근 위치로 지정하여 이동한다.
4. 마지막으로 최종 위치로 이동한다.



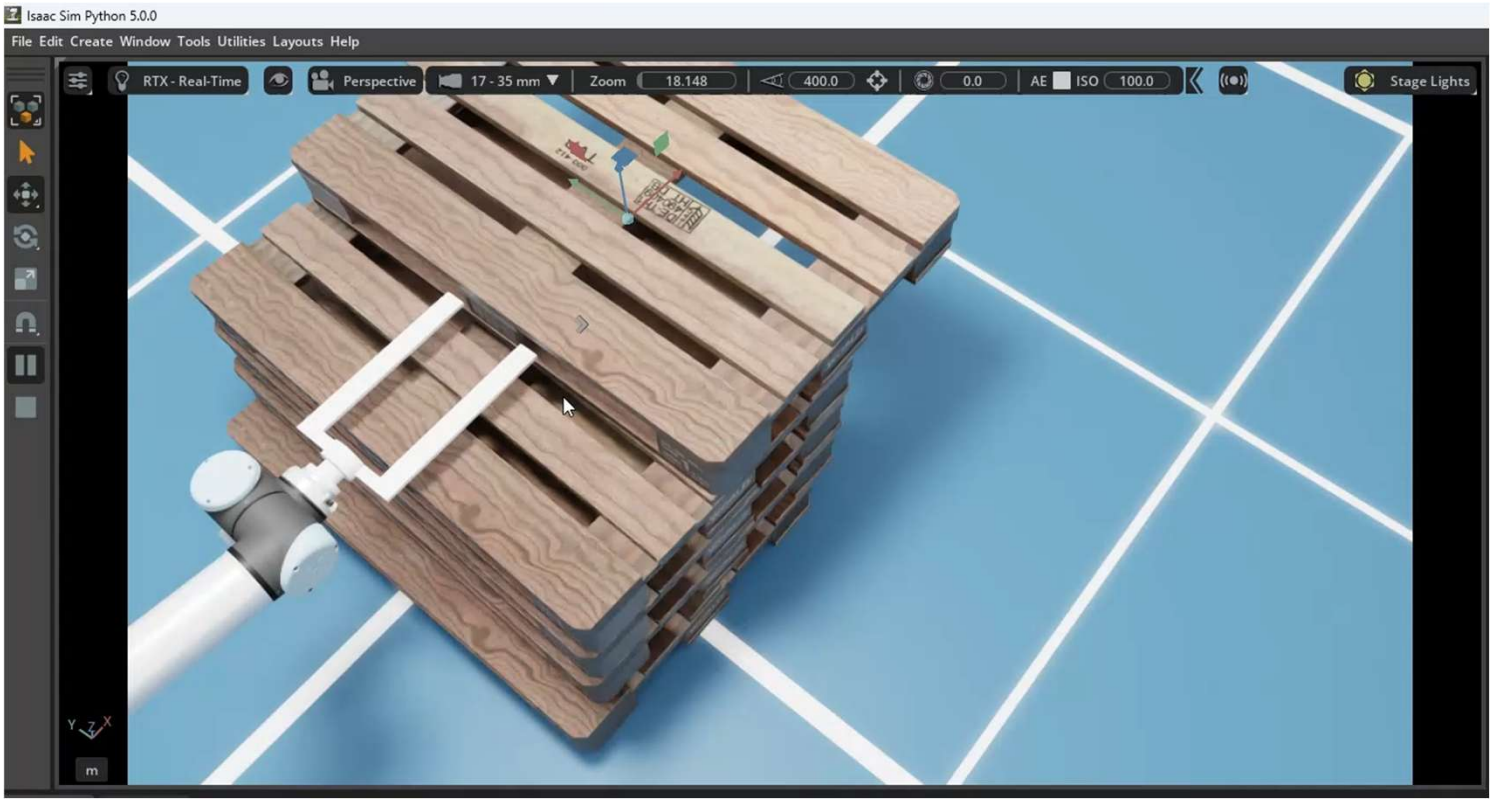
4. 로봇팔 로직

로봇 그리퍼 생성

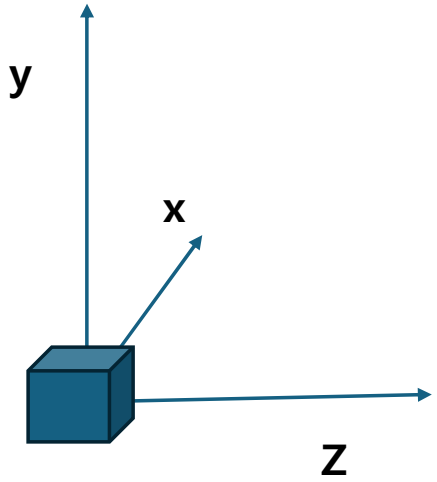


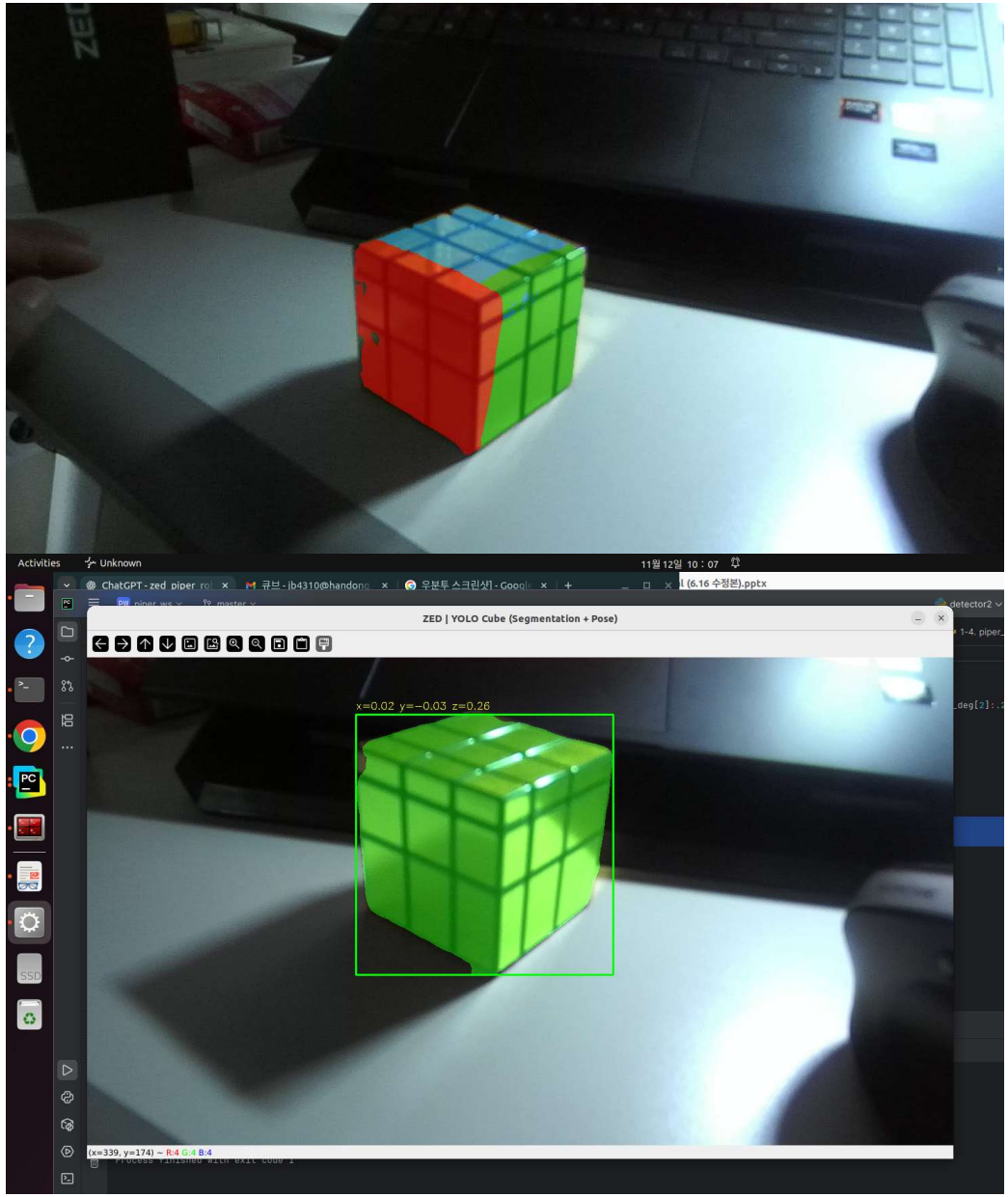
5. 문제점

팔렛 구멍에 안들어감



6. Zed Coordinate





6. Zed Coordinate

7. Isaac Sim 사용

DOPE, CenterPose 사용

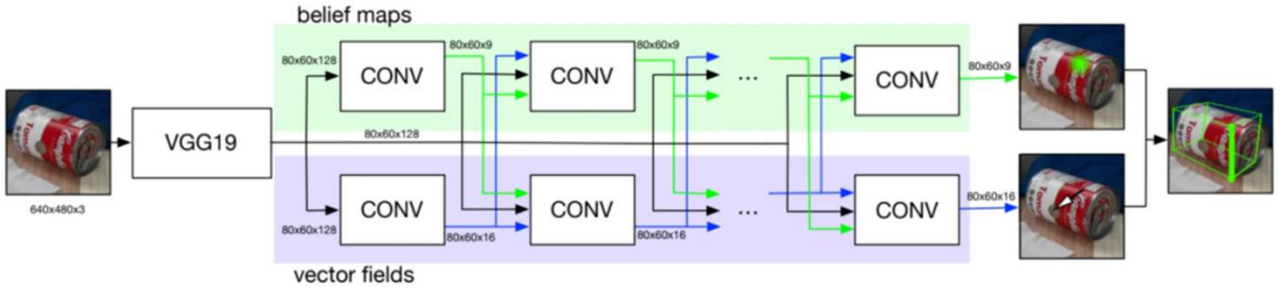
물체를 감싸는 큐브의 키 포인트를 찾는 딥러닝 모델

6DoF pose를 추정하기 위해 쓰인다.

DOPE는 ROS 연동해서 쓰기 위한 NVDIA 전용 모델

CenterPose는 DOPE에 비해 더 범용적으로 사용되는 모델이다.

둘다 example 존재



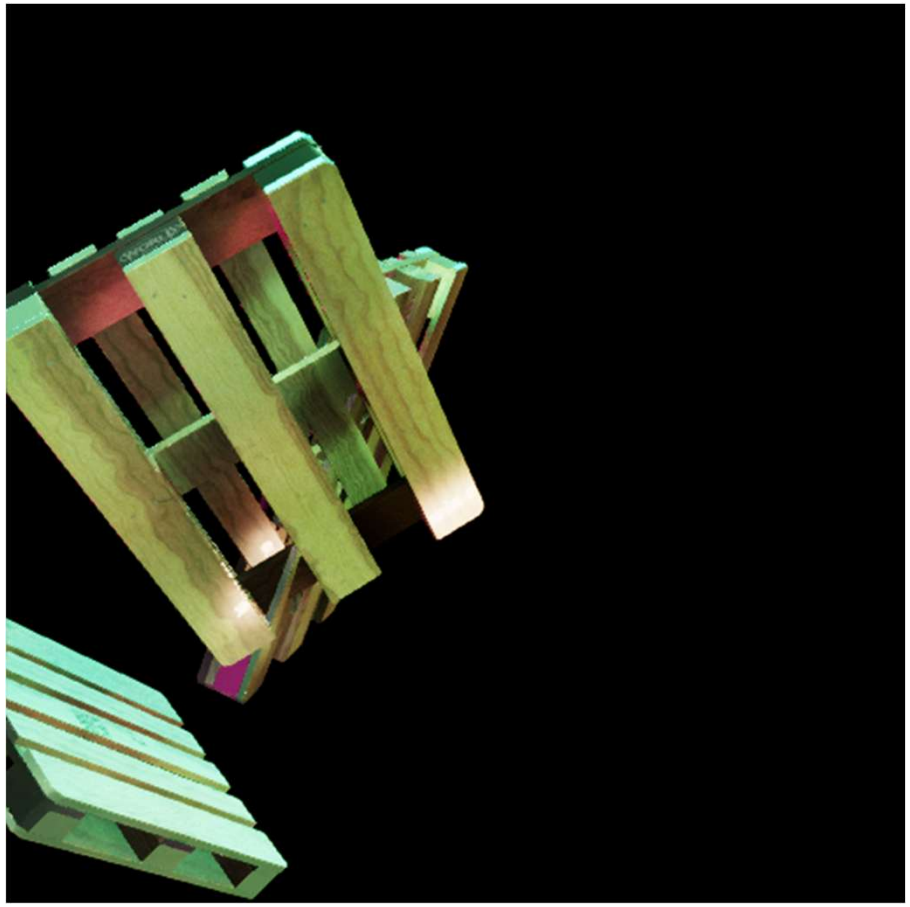
7. Isaac Sim 사용

아이삭심의 랜더링과 물리 엔진을 사용하여 현실과 비슷한 데이터셋을 만들 수 있다.

데이터 셋을 확보하기 어려운 환경에서 사용하기 적합하다.

현재 팔렛에 대한 이미지 데이터와 json 파일 생성 완료

CenterPose 학습을 위한 환경 구축 중



구매 할 것

8. Point Cloud 처리

물체 인식 후 bbox 안에 있는 포인트 클라우드

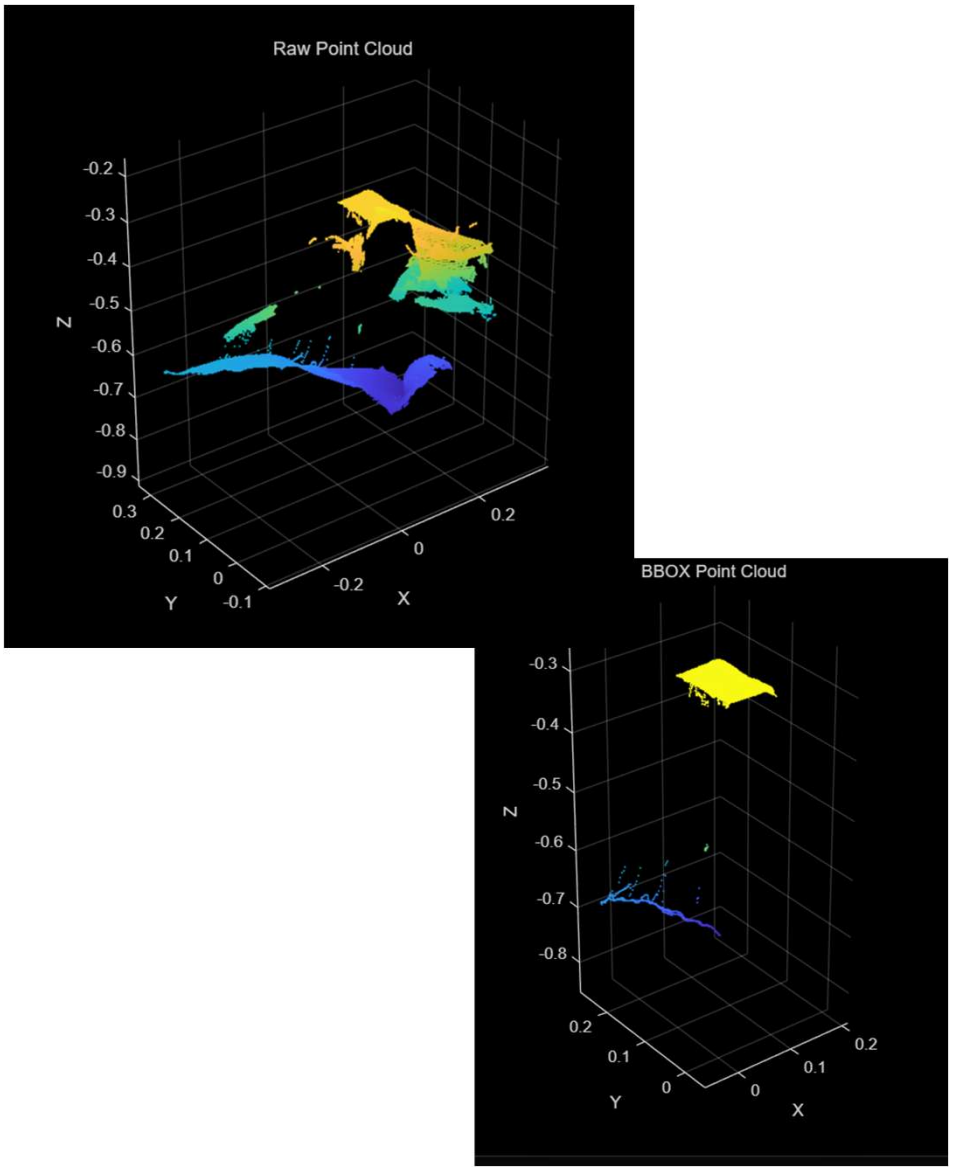
데이터 받아오기

DBSCAN 클러스터링을 통해 바닥이나 노이즈를

없애고 팔렛만 남기기

PCA로 얻은 주축 벡터 $\theta = arctan2(v_y, v_x)$으로

그리퍼 각도 구하기



8-1. DBSCAN

Density-Based Spatial Clustering of Applications with Noise

Eps : 반경

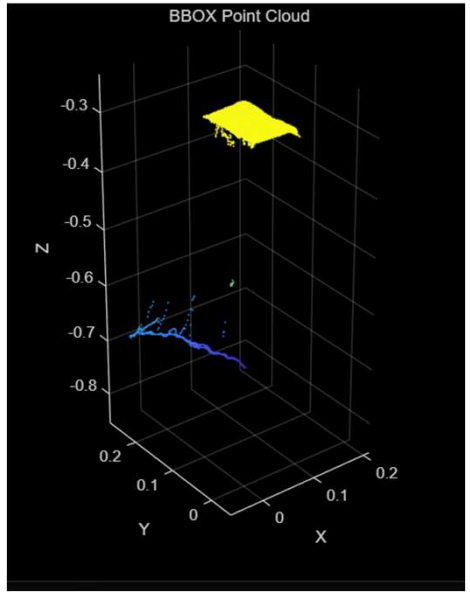
Min_points : 반경 안에 있어야 하는 최소 점 개수

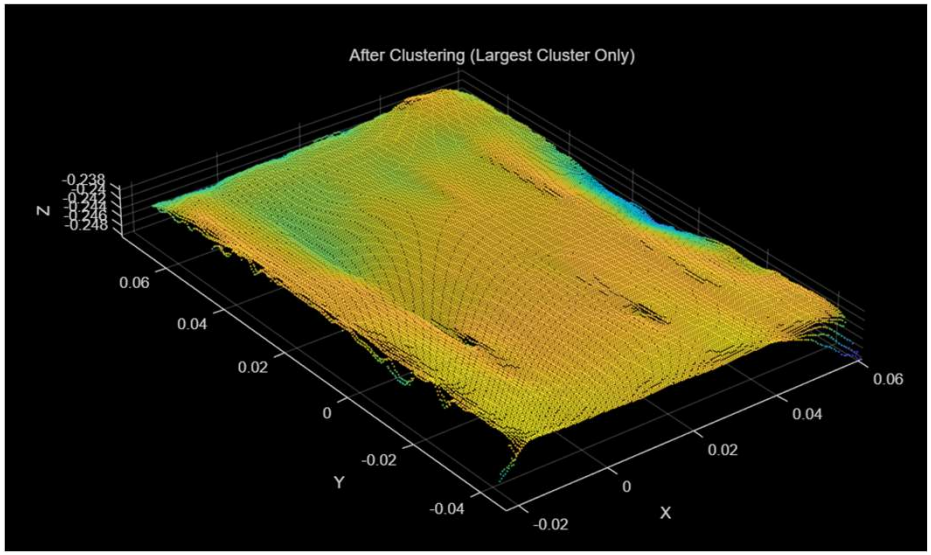
랜덤으로 선택한 점의 반경 안에 점의 개수가 min_points를 안 넘으면 -1로 처리

내 코드에서는

Eps: 0.005

Min_points : 100





8-2. PCA and RANSAC

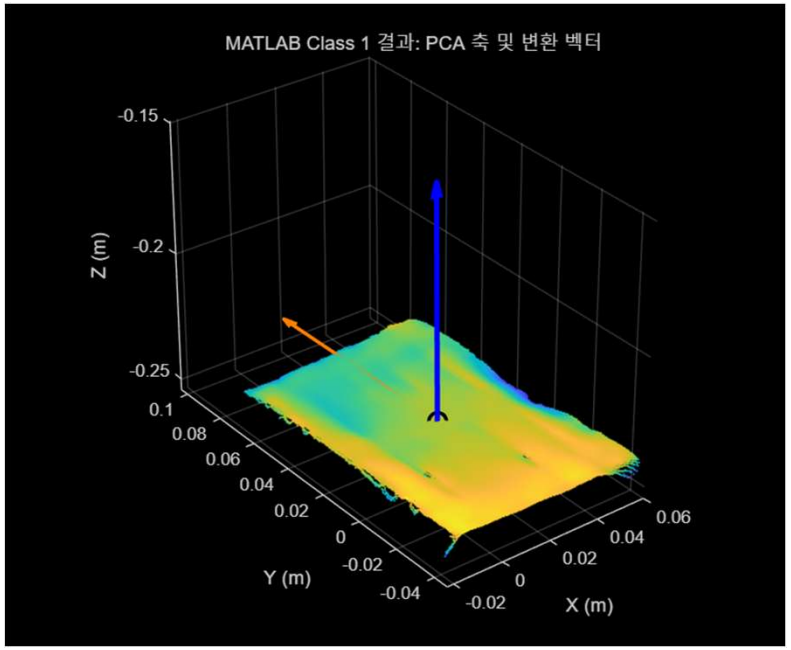
PCA : 분산 기반 주성분 방향 3축을 구한다

RANSAC : 포인트의 개수와 거리를 반복하여 보면서 평면을 피팅한다.

distance_threshold = 0.005

ransac_n=3

max_iterations = 1000



9. 좌표계 변환

ZED 뎁스 좌표계

X: 오른쪽

Y: 아래

Z: 전방

단위 meter

카메라 좌표계

X: 오른쪽

Y: 아래

Z: 전방

단위 pixel

그리퍼 좌표계

X: 전방

Y: 왼쪽

Z: 위

단위 mm

로봇 좌표계

X: 전방

Y: 왼쪽

Z: 위

단위 mm

No Need

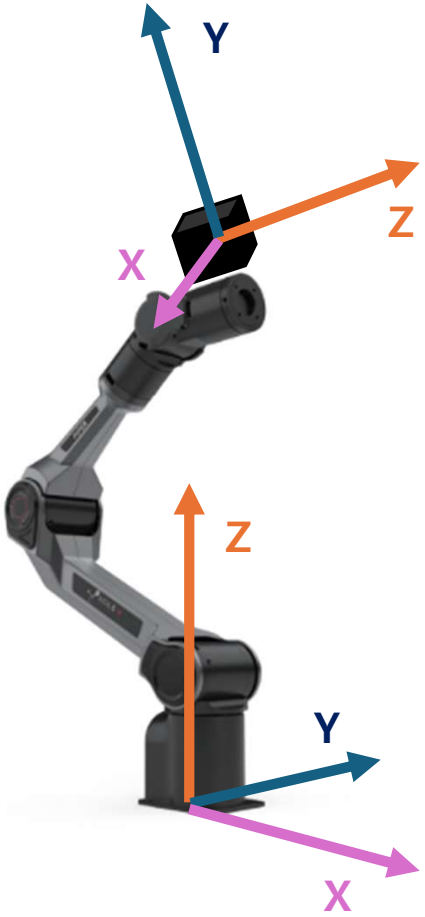
$$T_{cg} = \begin{bmatrix} R_{cg} & t_{cg} \\ 0 & 1 \end{bmatrix}$$

로봇이 제공하는 ee 포즈

$x_{ee}, y_{ee}, z_{ee}, r_x, r_y, r_z$

$$R_{cg} = \begin{bmatrix} 0.0653 & 0.9940 & 0.0867 \\ -0.9978 & 0.0656 & -0.0011 \\ -0.0068 & -0.0864 & 0.9962 \end{bmatrix}$$

$$t_{cg} = [-0.0408 \quad 0.0452 \quad 0.0497]^T$$



Result

10. DOPE(pose estimation)

Creating Realistic Dataset in many different environment

Based on VGG19, extract the feature

Belief Maps : heatmap of 8 conners and 1 center (= points)

Affinity Maps : combine points with vectors

SolvePnP : objectpoints, imagepoints, cameramatrix ->

Pose

10. DOPE Result

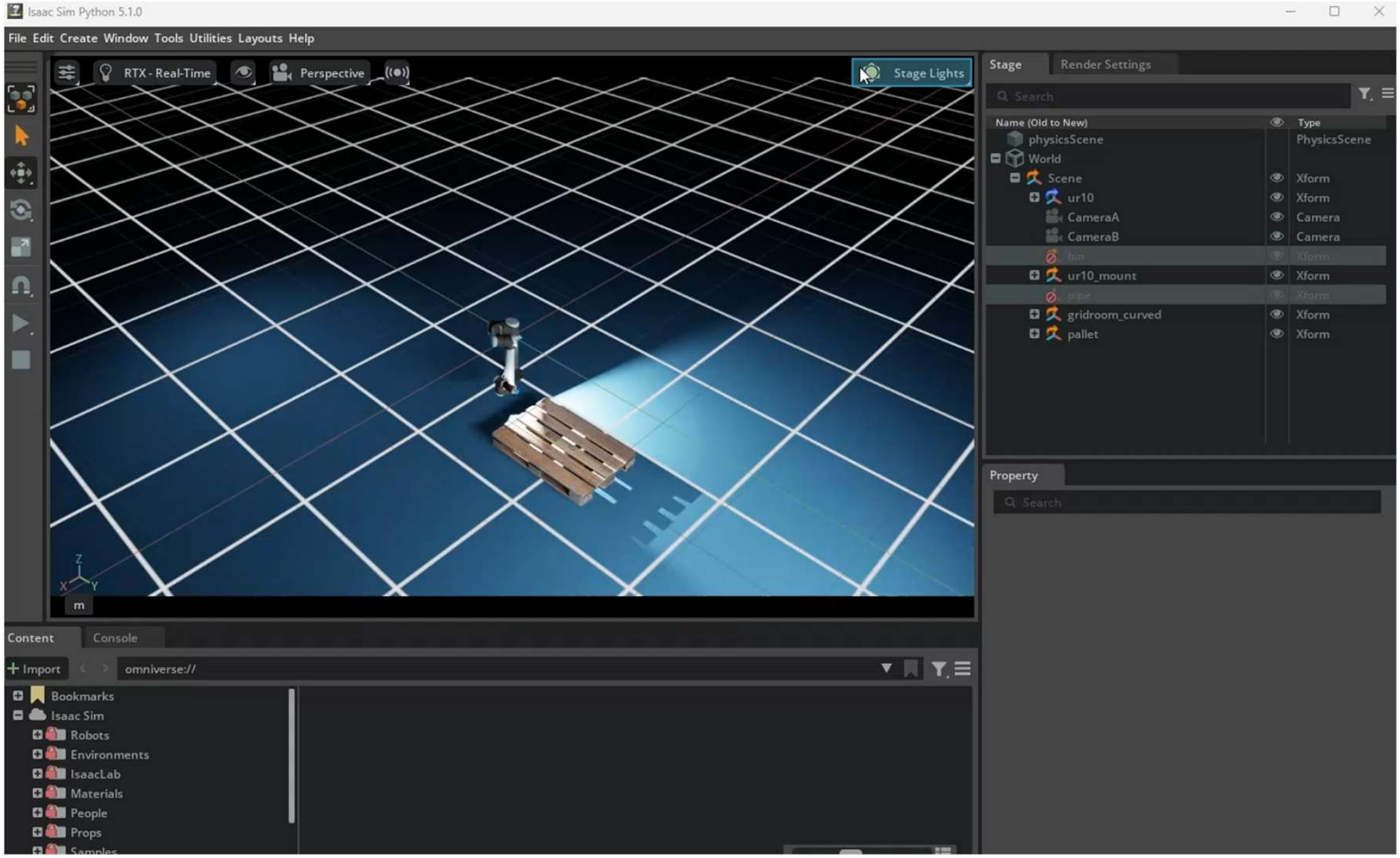
다음주 계획

13 – DOPE 모델 강화, 시뮬레이션 구현

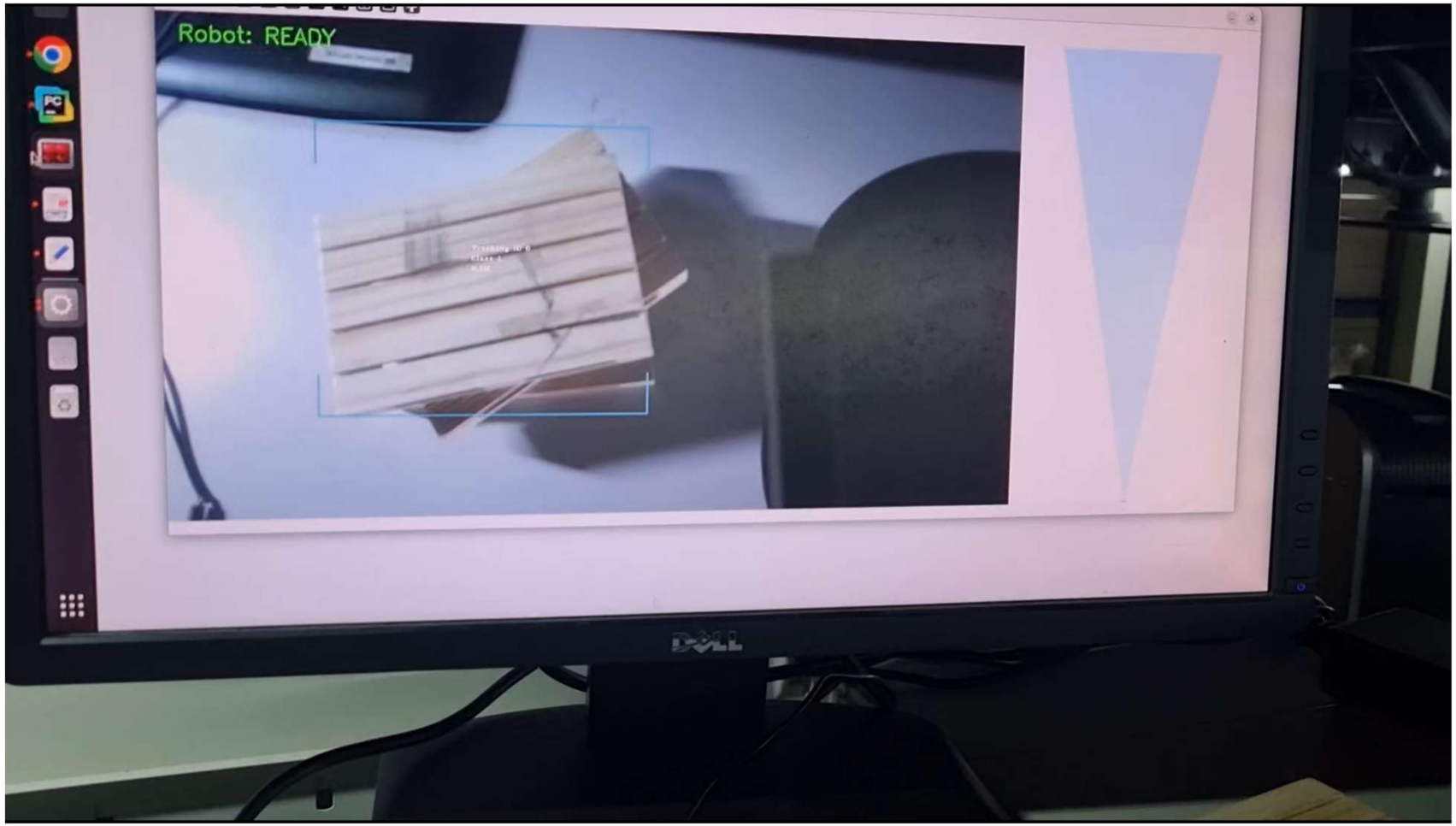
14 – DOPE 모델과 포인트 클라우드 처리 비교

15 – 최종 발표

시뮬레이션 픽엔플레이스 완료



실제 픽엔플레이스 완료



DOPE 학습 데이터 개선

배경 수정

포인트가 7개 이하일 경우 제거

-> 학습 중

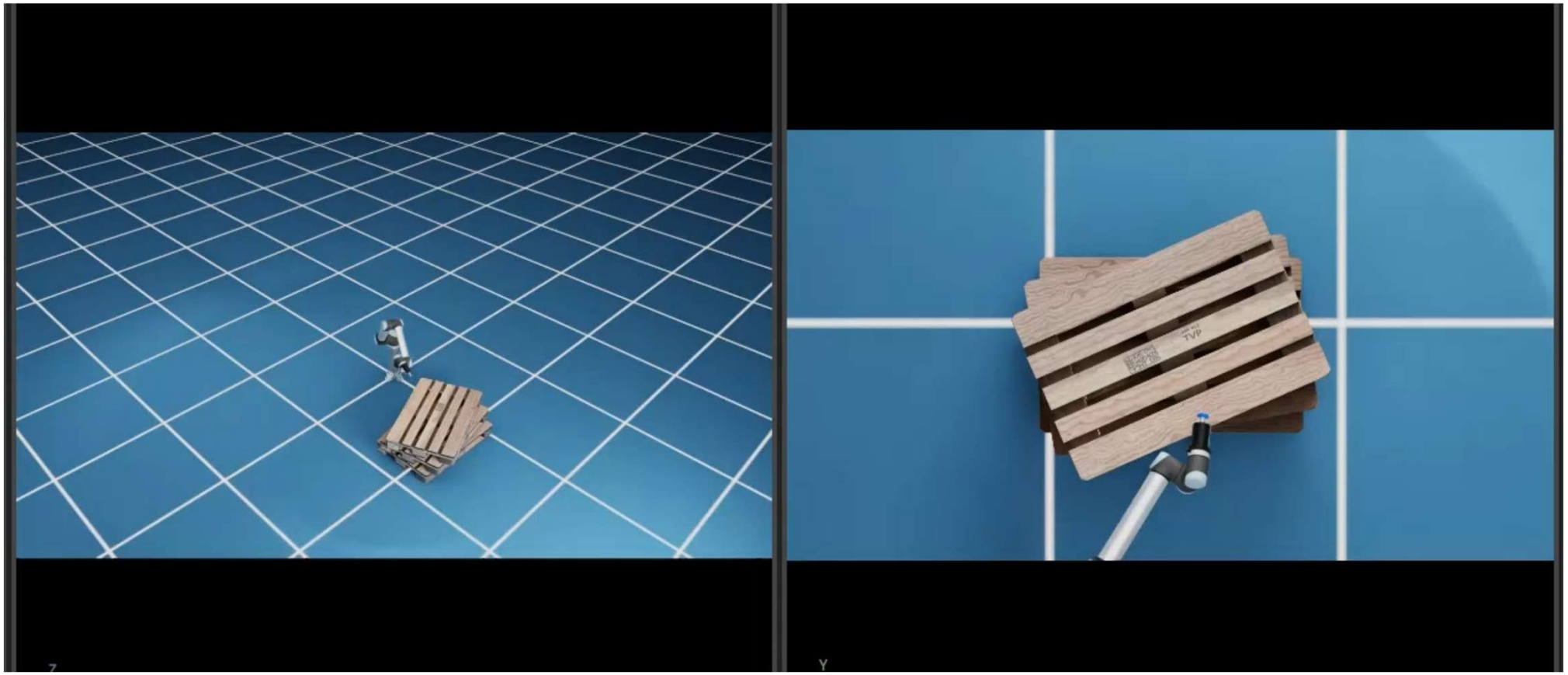
다음주 계획

DOPE 성능 확인

시뮬레이션 픽엔플레이스 구현

성능 비교 : 각도 예측, 속도

시뮬레이션 마무리



포인트 클라우드 처리 기법 성능

시뮬레이션 상에서 100번 동안 -80도에서 80도 랜덤하게 준 팔

렛의 자세와 포인트 클라우드 응용 기법으로 구한 팔렛 자세

비교

True angle과 pca angle, 그리고 error 값을 csv파일로 저장

	A	B	C	D
1	iteration	true_angle	pca_angle	error_deg
2	0	-4.417614	-4.708082	0.290467858
3	1	30.687013	34.140589	3.453575763
4	2	37.434597	39.528883	2.094285662
5	3	-68.20791	-65.04204	3.165868146
6	4	-76.24711	-74.35825	1.888862535
7	5	65.023359	61.269082	3.754277351
8	6	44.854046	44.696315	0.157730481
9	7	26.578955	30.101155	3.522200511
10	8	-4.912617	-5.187389	0.274772098
11	9	76.299176	74.123392	2.175784013

Error = ±2.308°

DOPE 자세 추정 성능

저장된 사진과 json 파일가지고 학습한 모델로 포인트를 찾아

실제 포인트 위치와 예측된 포인트 위치 차이를 보았다.

Error = ±89.53 pixel

예측된 포인트 가지고 자세각 구하는 로직 만들어야 함

filename	mean
004007.pn	17.0701561
004008.pn	9.843385696
004009.pn	165.829422
004010.pn	17.99238205
004011.pn	164.3473206
004012.pn	48.6350975
004013.pn	36.83589935
004014.pn	70.78668213
004015.pn	294.4177246
004016.pn	246.4674225
004017.pn	8.033243179
004019.pn	38.35742569
004020.pn	19.33403969

Poster Outline

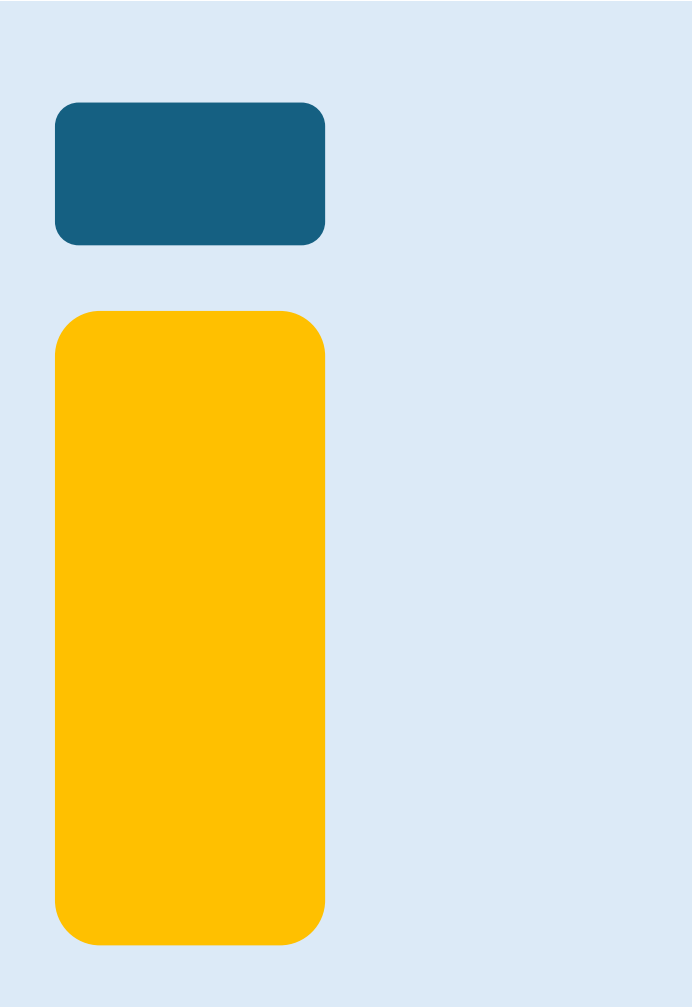
Introduction

1. Background : 산업 현장에 부족한 데이터와 실험을 물리 엔진과 하이 레벨 렌더링이 적용된 아이삭심에서 구현함으로써 학습성능을 높이고 시간과 재정을 아낄 수 있다. (+ 아이삭심 설명)
2. Object : 공사 현장에 랜덤하게 쌓여있는 팔렛을 인식하여 정렬되어 pick and place 진행
3. Methodology : Point Cloud Based Vector Estimation & DOPE pose estimation

Poster Outline

Point Cloud Based Vector Estimation

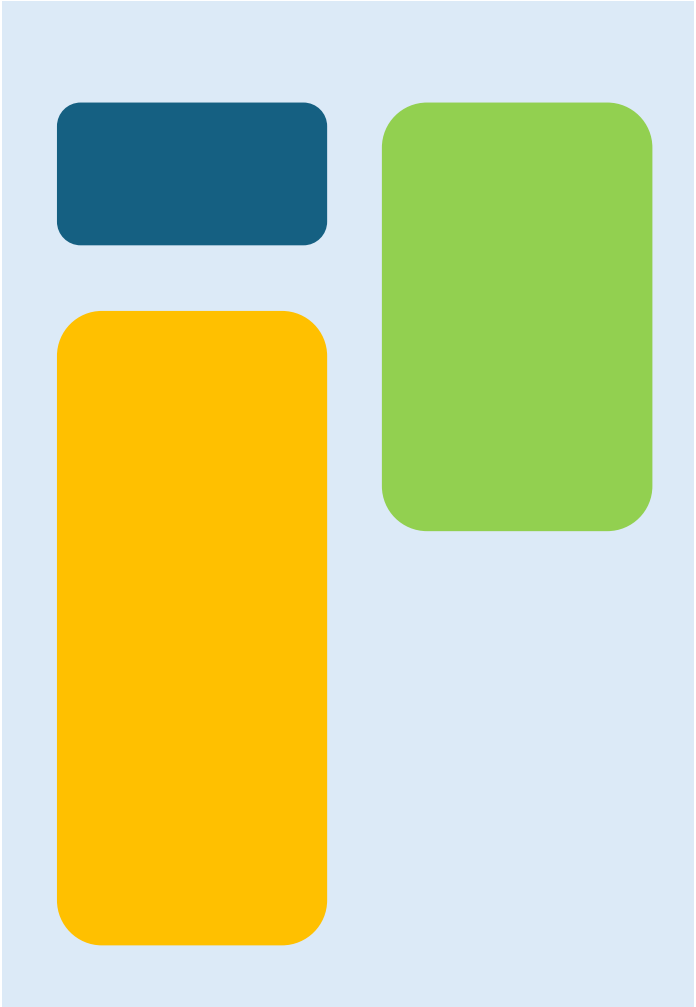
1. 시뮬레이션 세팅 설명 : 카메라 위치, 팔렛(물리 엔진), 로봇팔
2. 방법 설명 : YOLOV8로 인식, Bbox안에 있는 포인트 클라우드 DBSCAN, RANSAC 면피팅, PCA로 사이드 벡터 추정, 로봇팔 작동
3. 전체 환경 사진 + 단계별 사진
4. 결과



Poster Outline

DOPE Pose Estimation

1. DOPE 모델 설명 (+ 아이삭심과의 호환성)
2. Train + Validation dataset
3. 결과



Poster Outline

실제 구현

1. 구현 환경 설명
2. 로직 설명
3. 결과 사진

결론 + future study