Measuring Image Distance(MIS) in meter unit

6조 박성호, 박종엽, 손도희, 김아영
팀장 : 박성호(010-4147-5331, relilau00@gmail.com)

목차

1. 구성원 소개

순번	이름	학번	역할	전화번호	이메일
1	박성호 (팀장)	201611263	Feature 선정 및 추출 Model/Merger 개발, 평가 진행, 보고서 작성	010-4147-5331	relilau00@gmail.com
2	김아영	201611255	논문 리뷰 및 분석, Data 검증/수집	010-4541-7859	kay511@naver.com
3	박종엽	201511263	Web 서비스, API 서버, Image processor 개발	010-9156-8254	whdduq218@gmail.com
4	손도희	201714165	논문 리뷰 및 분석, Data 검증/수집, Web 서비스 개발	010-4249-9880	dhson987@konkuk.ac.kr

2. 작품 소개

개발 동기

Object 기반의 Distance 예측 연구들과 다르게, 이미지의 pixel 단위로 distance를 예측하여 이미지의 모든 정보를 활용해야 하는 타 연구활동들에 도움을 주기 위함

작품 설명

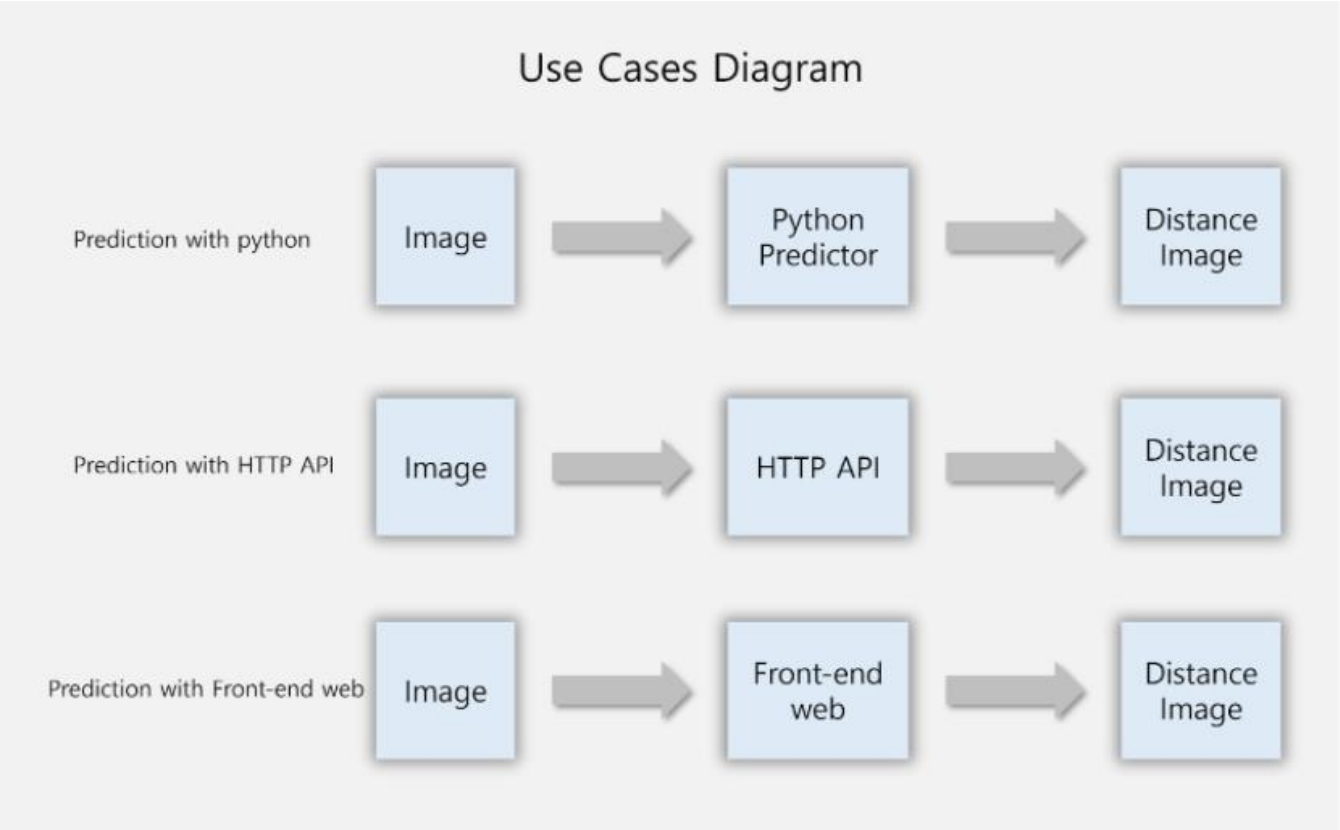
배포 내용: predictor와 해당 모델 학습을 위한 train 코드, HTTP 1.1 RESTful API, Front-end web

배포 기간: python 코드는 무기한 배포 예정, API/Web 서버는 졸업프로젝트 심사 기간 동안 가동될 예정

배포처: Github Repository, 개인 서버

배포방법: Github Repository. HTTP 1.1 RESTful API, Front-end web

전체적인 개념도


Use Cases Diagram
Prediction with python
Image
Python Predictor
Distance Image
Prediction with HTTP API
Image
HTTP API
Distance Image
Prediction with Front-end web
Image
Front-end web
Distance Image

3. 핵심 문제 정의 및 해결방안

Key issue

Measuring distance of image's all pixels in meter unit.

Solution

1. Train Process
2. Merge Process
3. Check RMSE

Train Process

1. Make raw data
2. Make train data
3. Make model
4. Load data
5. Train

Reference

1. Learning Object-Specific Distance From a Monocular Image(2019)

- ROI Pooling을 활용하여 Distance를 학습하는 CNN model 구조에 대한 아이디어 얻음
- https://arxiv.org/pdf/1909.04182.pdf

2. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks(2015)

- ROI Pooling의 구현에 대한 method를 일부 채용
- feature map을 만드는 방법론 채택
- https://arxiv.org/pdf/1506.01497.pdf

3. NYU Depth V1

- 뉴욕대학교에서 공개 배포하는 Meter 단위의 distance와 RGB 이미지가 매칭된 데이터 셋 사용
- https://cs.nyu.edu/~silberman/datasets/nyu_depth_v1.html

4. Mega Depth: Learning Single-View Depth Prediction from Internet Photos(2018)

- 임의의 이미지로부터 depth를 예측하는 모델을 depth predictor로써 사용
- https://www.cs.cornell.edu/projects/megadepth/

Train Process

Make raw data

Set I matrix as image set of NYU Depth V1 data set.

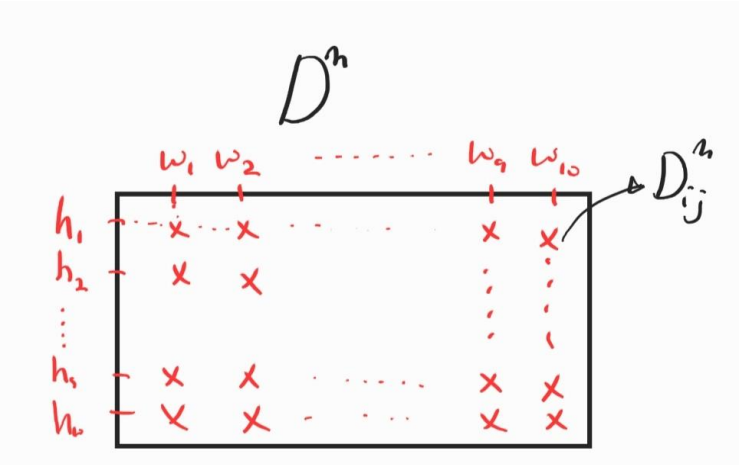
Set I^n *as* n^{th} *image of I matrix.*

Set D^n *which is one-to-one corresponding 2D distance information of* I^n.

Set 10 points on each width and height of D^n.

Set D_{ij}^n *as (i, j) distance information of* D^n.

Save D_{ij}^n *and (i, j) in memory.*

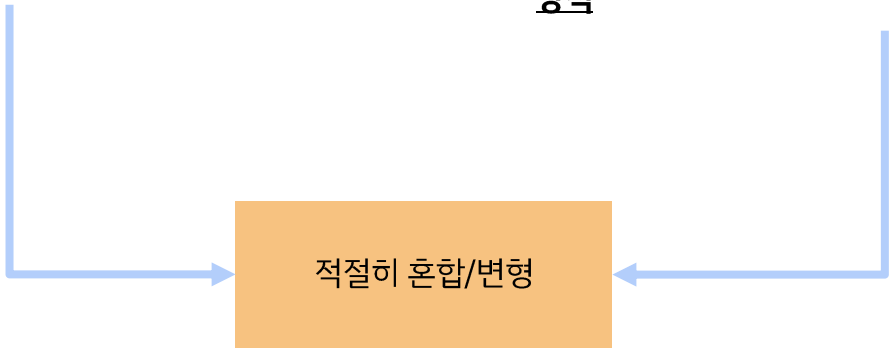


Train Process

Make model

"Learning Object-Specific Distance From a Monocular Image (2019)"(이하 LOSD 논문)에서 **fully connected layer를 쌓아서 Distance를 학습하는 process**

"Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks(2015)"(이하 Faster R-CNN 논문)에서 **pre trained model로부터 feature map을 얻는 방식**



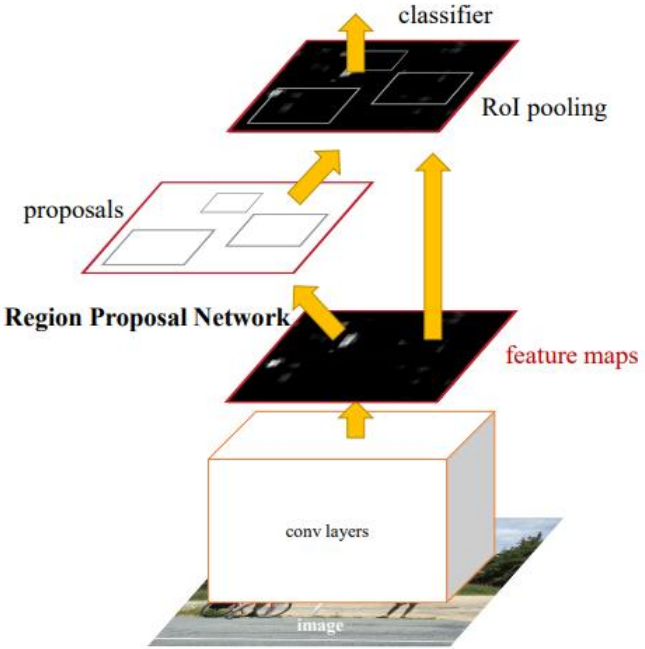
Train Process

Make model – Make feature map

CNN(vgg, resent) → Feature map

(800, 800, 3) → (50, 50, 512)

sub sampling ratio = 1 / 16



Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks(2015)

Train Process

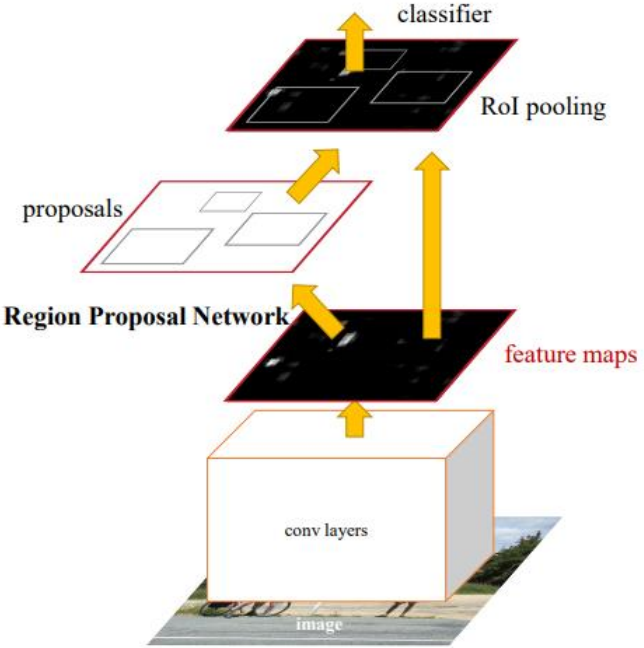
Make model – Make feature map

(800, 800, 3) → (50, 50, 512)

Set S_{cnn} as output size of conv layer.
Set S_{ssr} as reduced image size by sub sampling ratio
Set S_{fm} as feature map size.

$S_{fm} = (S_{ssr}, S_{ssr}, S_{cnn})$

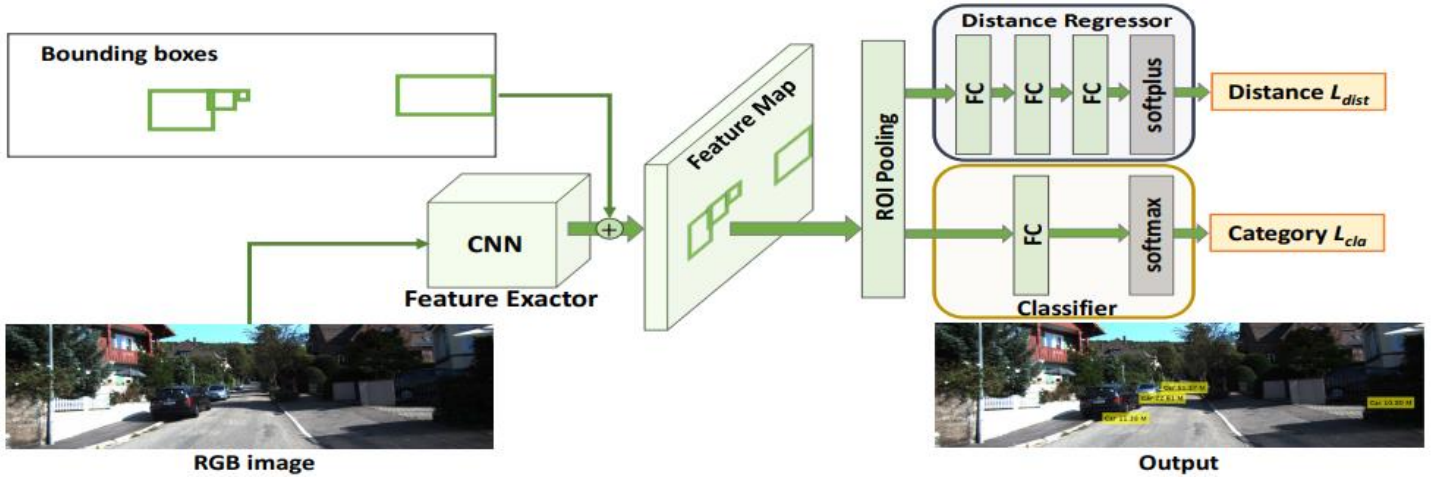
Used CNN = VGG-11, Mobilenet_V3_small
S_{cnn} *of VGG-11= 512*
S_{cnn} *of Mobilnet_V3_small =48*



Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks(2015)

Train Process

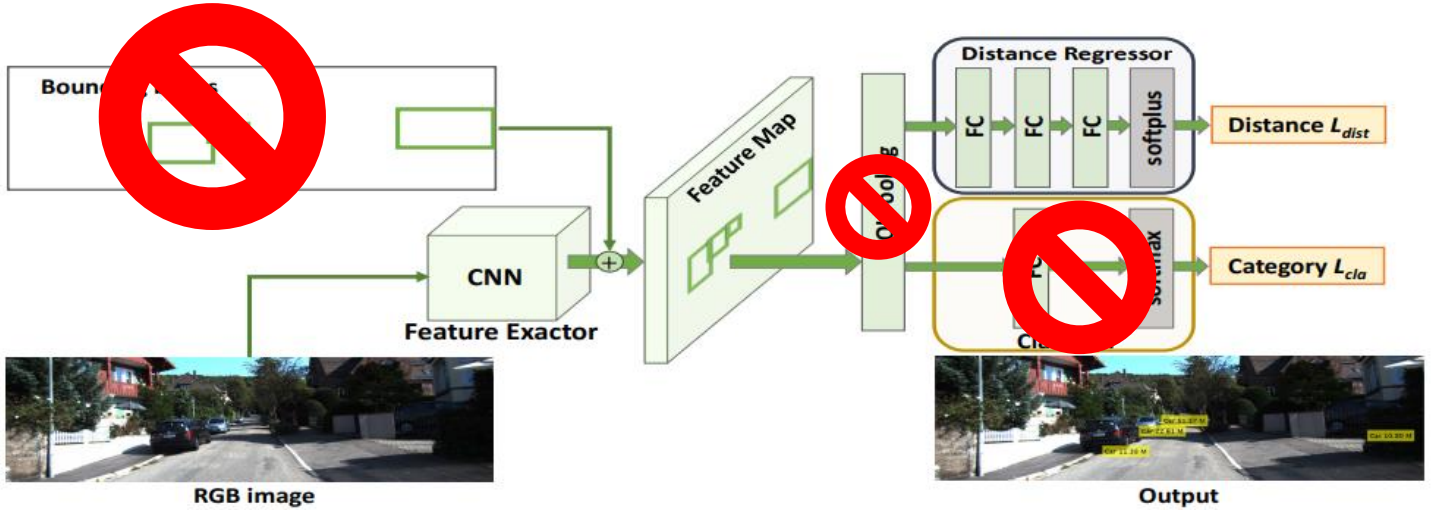
Make model – Distance regressor



Learning Object-Specific Distance From a Monocular Image (2019)

Train Process

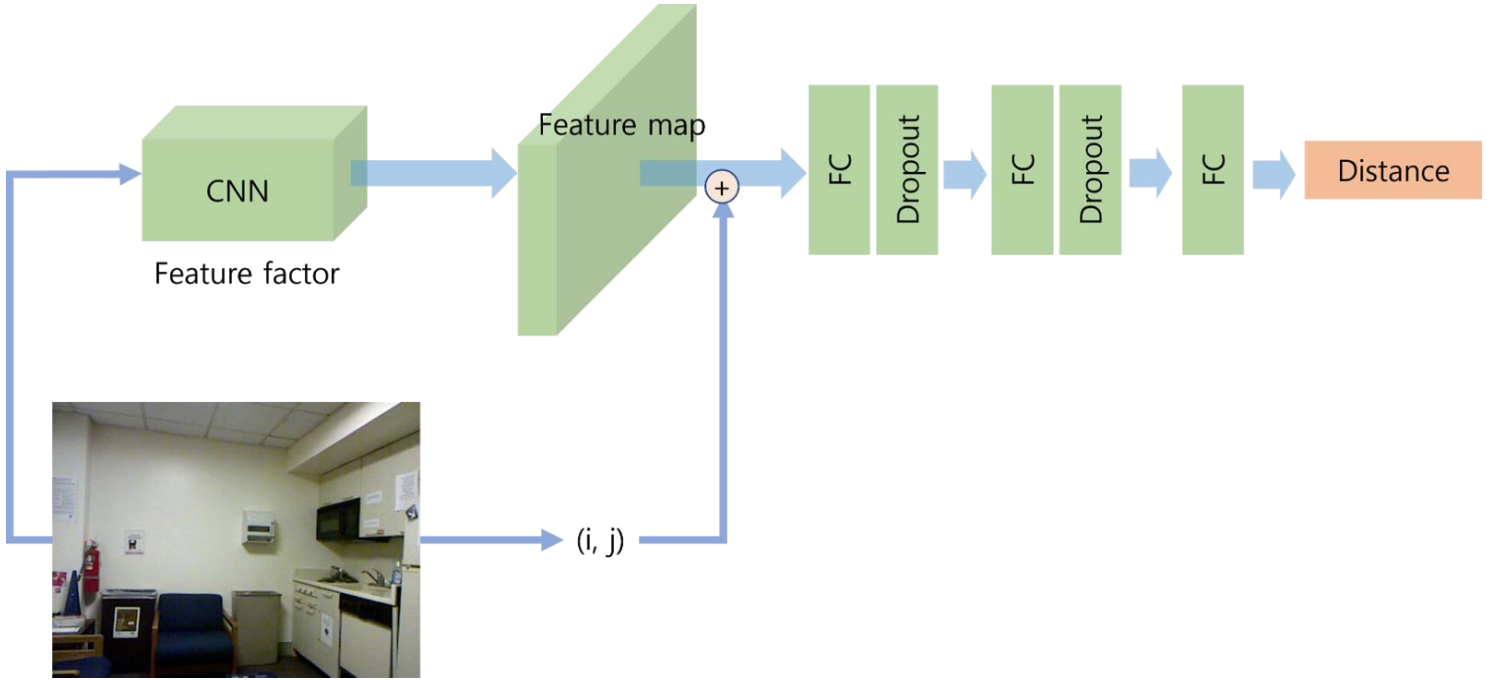
Make model – Distance regressor



Learning Object-Specific Distance From a Monocular Image (2019)

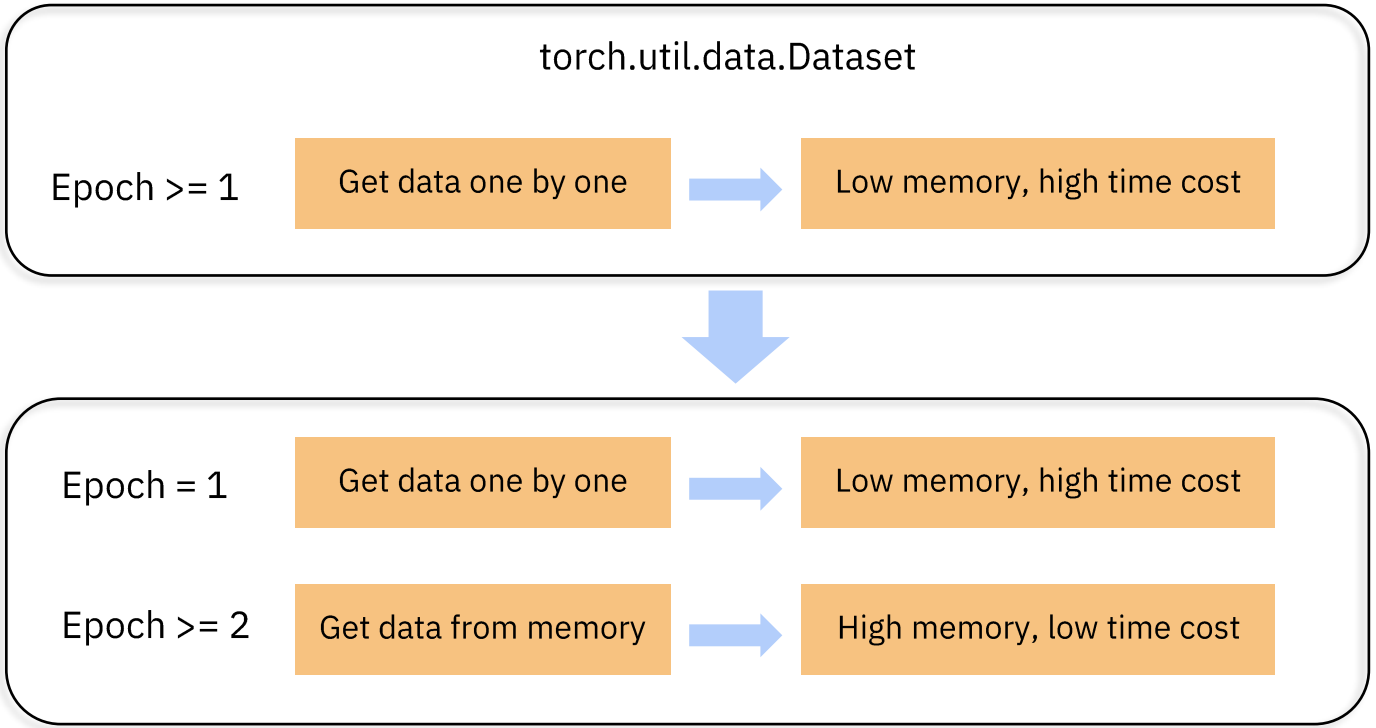
Train Process

Make model – Distance regressor



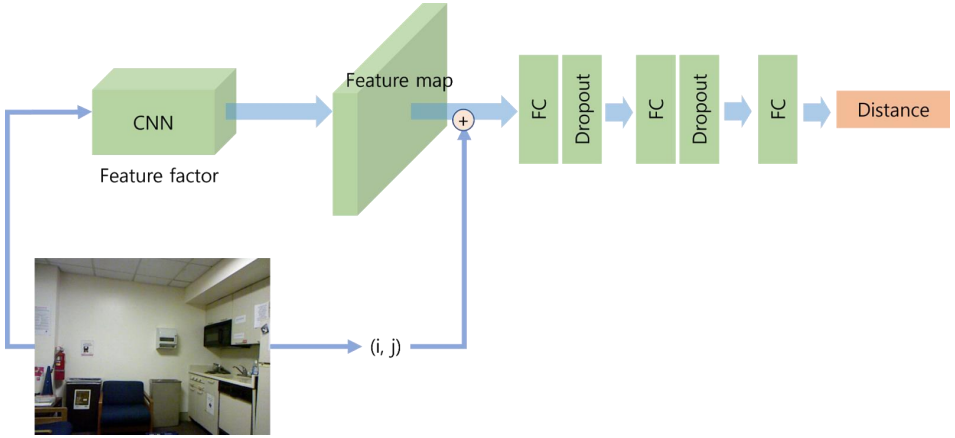
Train Process

Load data



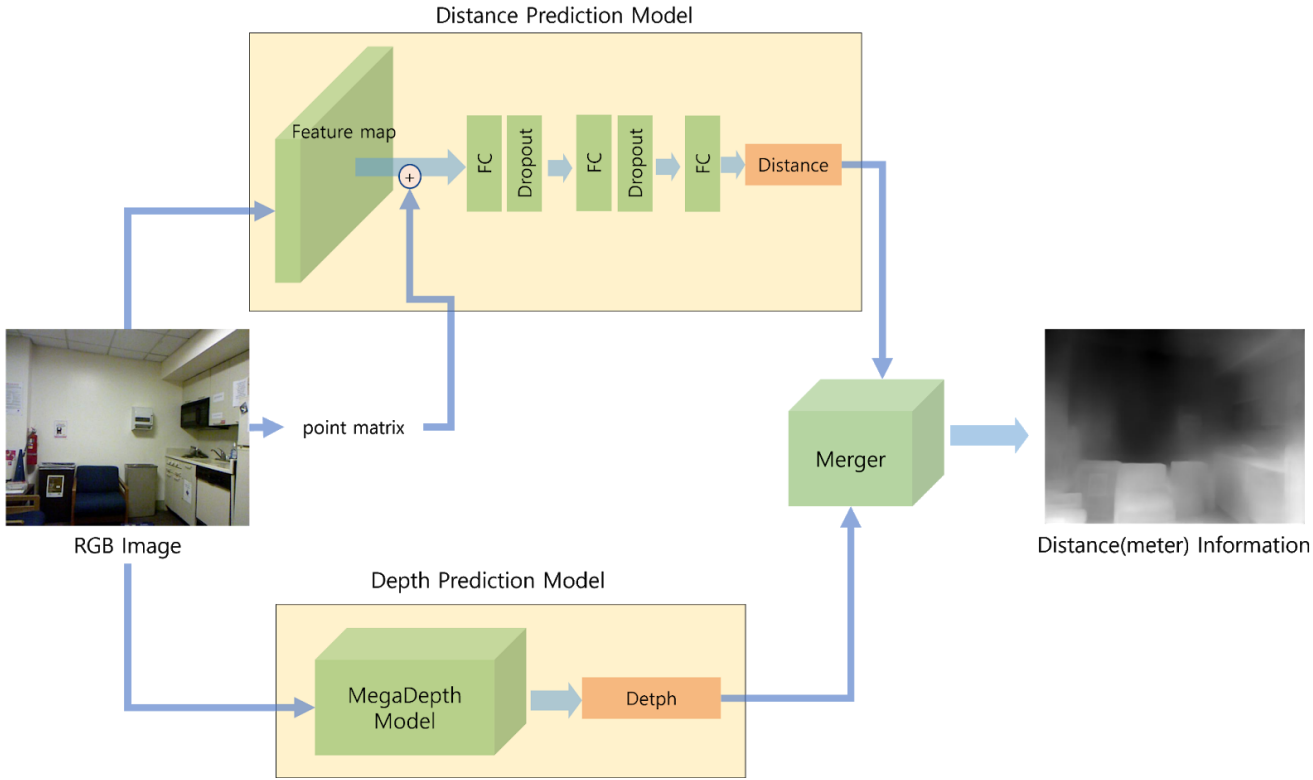
Train Process

Train



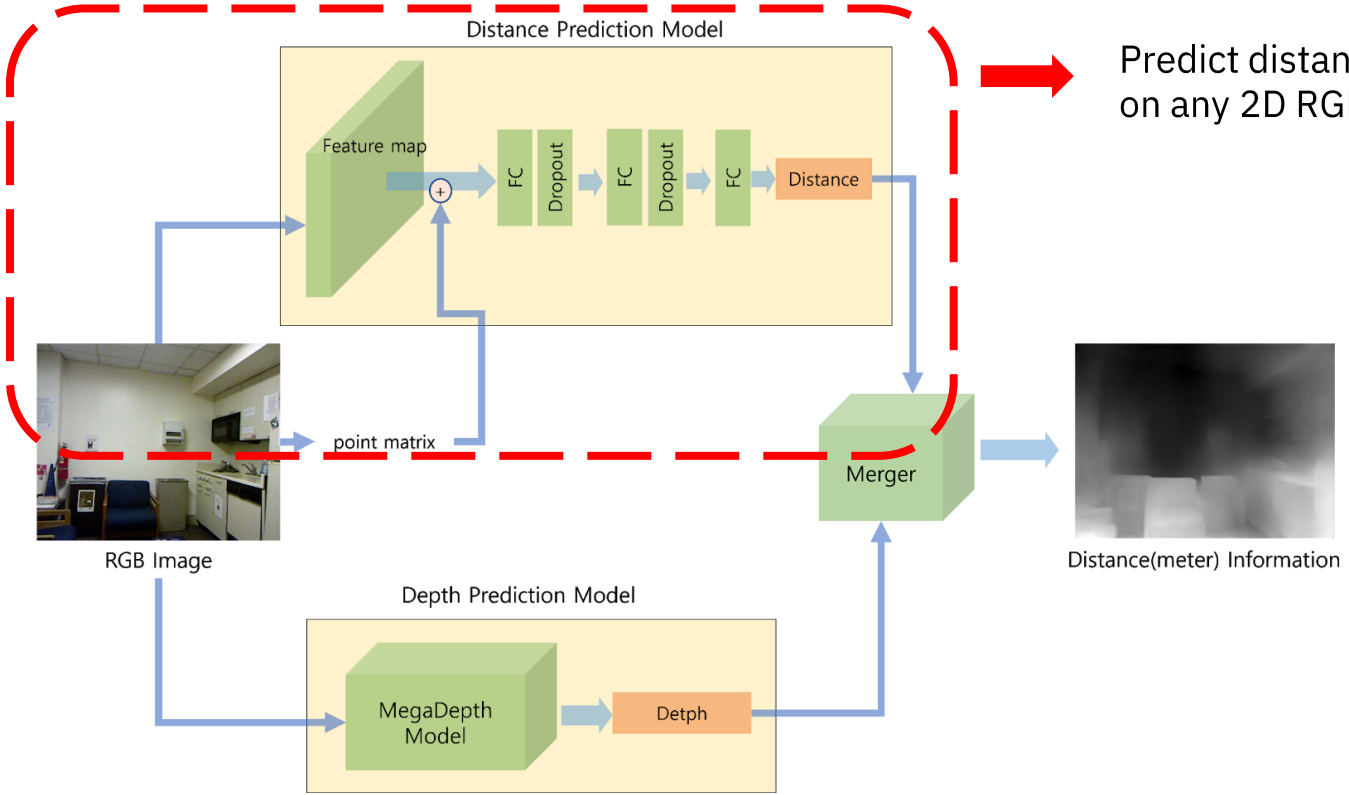
Feature map extractor = Vgg11, Mobilenet_v3_small → Predict distance of (i, j)

Merge Process



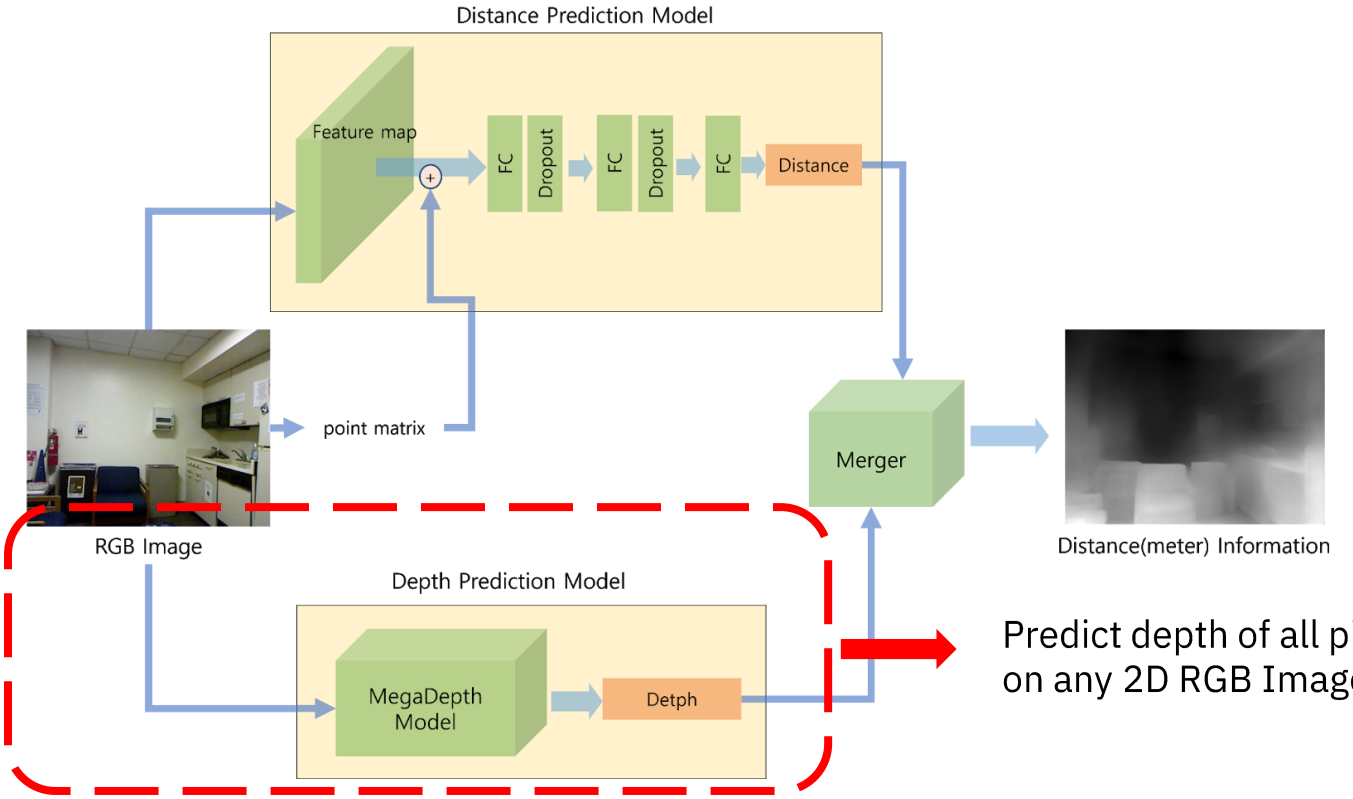
Merge Process

Predict distance



Predict depth

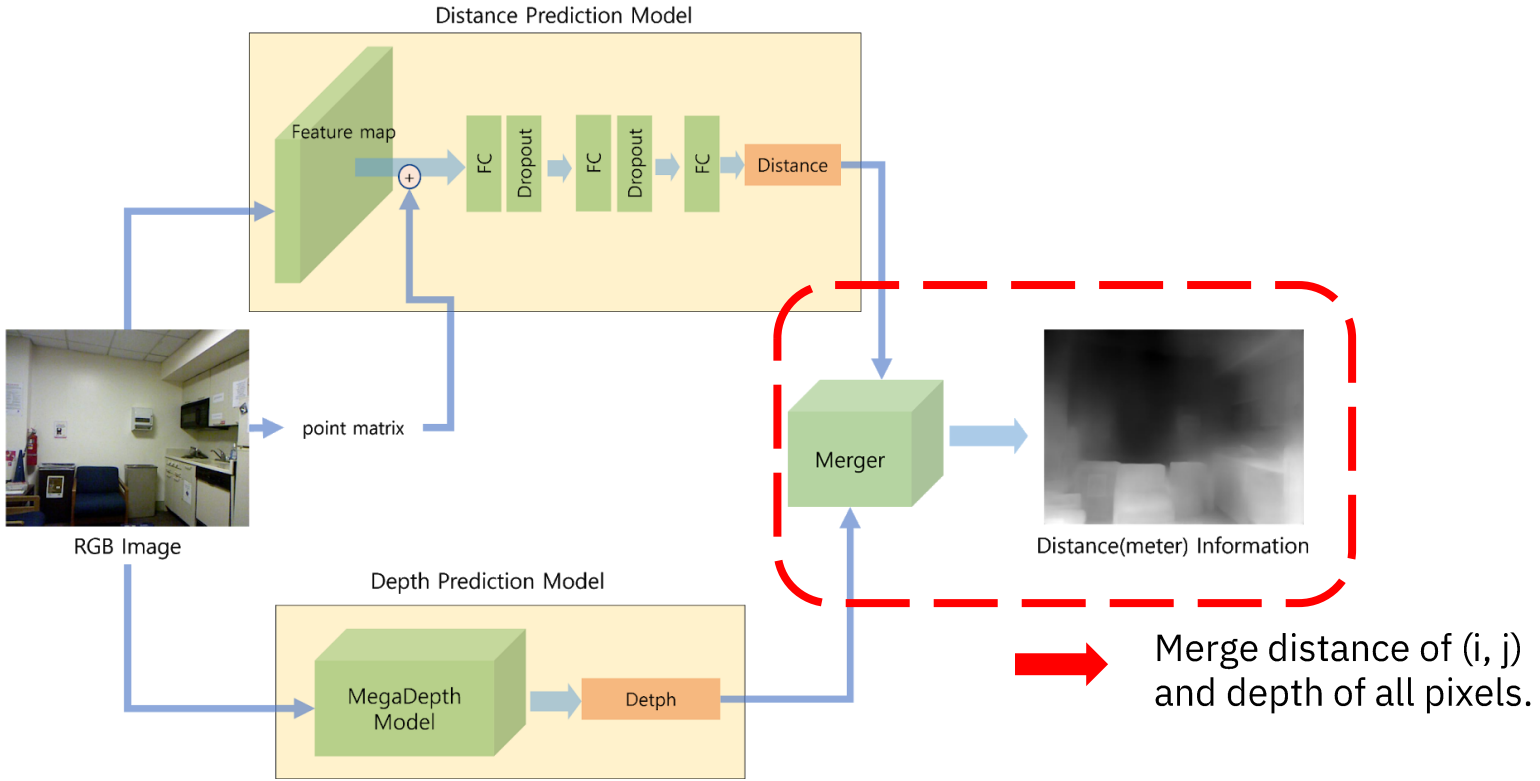
Merge Process



Predict depth of all pixels on any 2D RGB Image

Merge

Merge Process



Merge Process

Merge - set ratio - median order

Set predicted distance and predicted depth using median order.

Set prediction ratio as $\frac{predicted\ distance}{predicted\ depth}$

Set D_i *as distance matrix predicted by merger.*

Set D_e *as depth matrix predicted by MegaDepth model.*

$D_i = D_e \odot$ *prediction ratio*

Merge Process

Merge - set ratio – statistics

Set predicted distance and predicted depth using statistics method.

Mean, median, standard deviation.

Set prediction ratio as $\frac{predicted\ distance}{predicted\ depth}$

Set D_i *as distance matrix predicted by merger.*

Set D_e *as depth matrix predicted by MegaDepth model.*

$D_i = D_e \odot$ *prediction ratio*

프로그램 정확도 예측

RMSE(Root mean square error) with predicted result and real distance.

기존 논문과 비교

Advantage

넓은 범위에 대한 distance 데이터 확보 가능

넓은 시야에 대한 분석할 때 유리

Disadvantage

사건, 사고 중심의 탐지 프로그램에 적용 어려움

인간과 같은 방식의 시야가 아님

4. SW 설계

요구사항

기능 요구사항

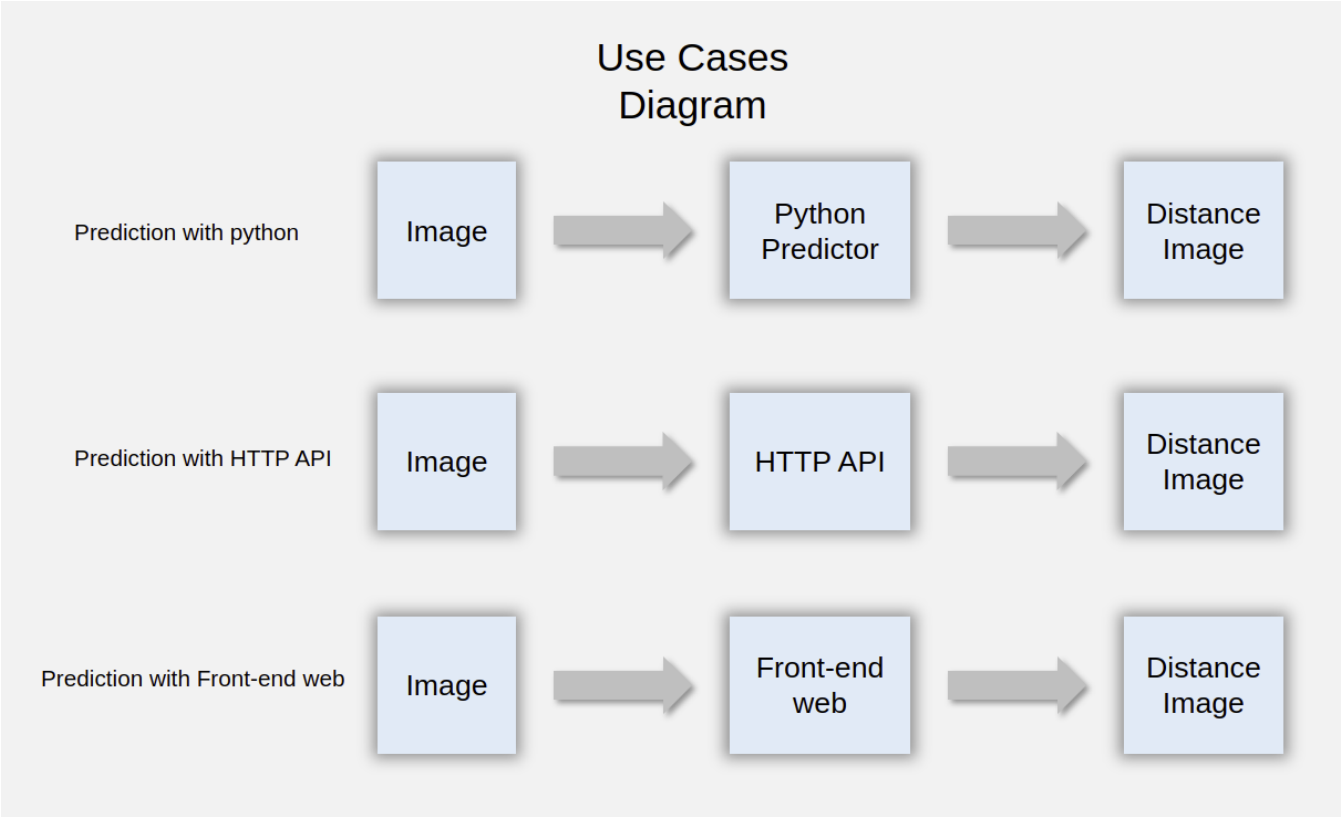
1. 임의의 RGB이미지에 대해서 pixel 단위로 meter 단위의 거리를 예측할 수 있음.
2. Predictor의 RMSE가 4를 초과하지 말아야함. (NYU Dataset의 거리 범위가 0 ~ 4)

→ Resolve with MIS model

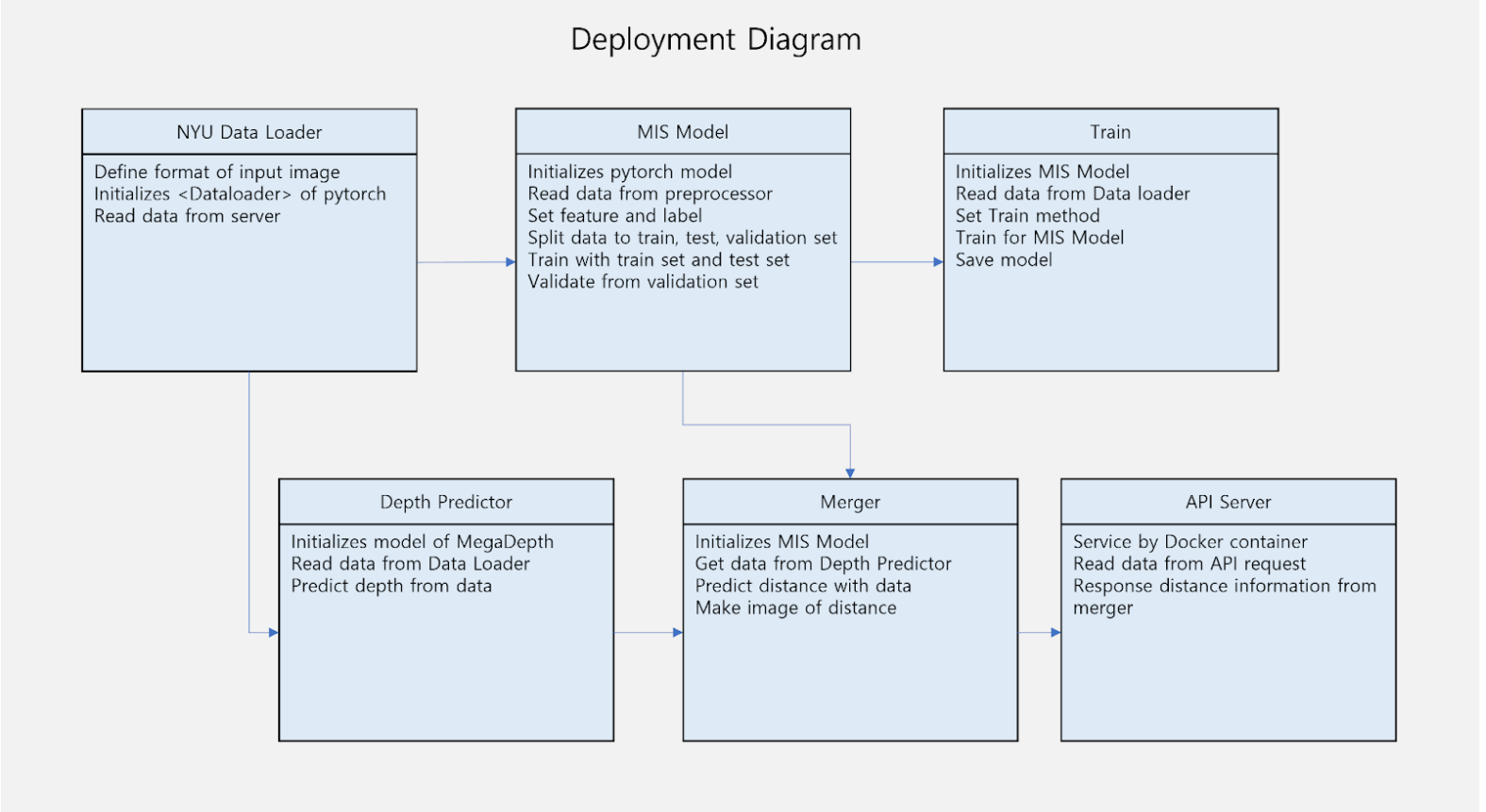
비기능 요구사항
- 임의의 사용자가 쉽게 프로그램에 접근 가능해야함.

→ Resolve with API Server, Front-Web

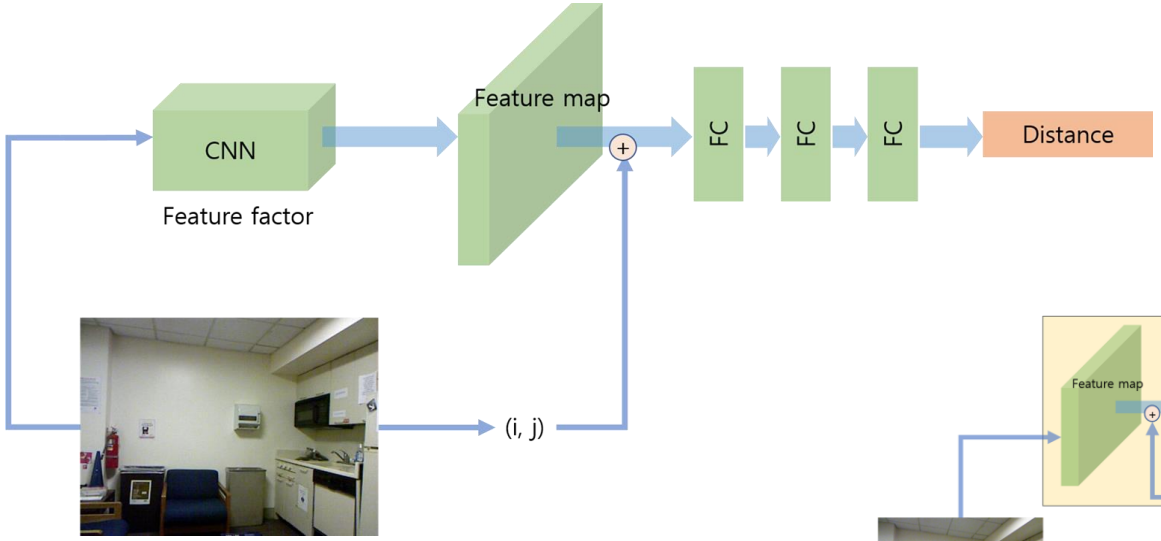
Use cases Diagram

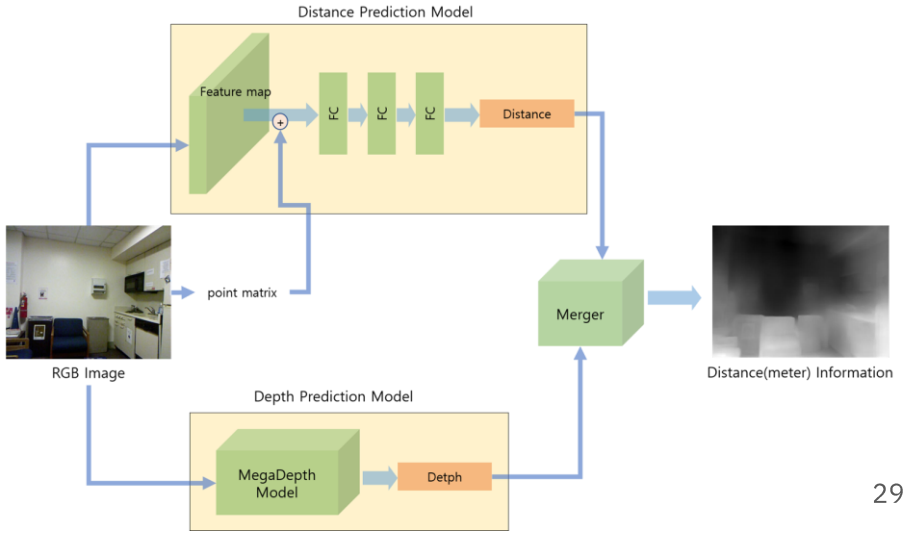


Deployment Diagram

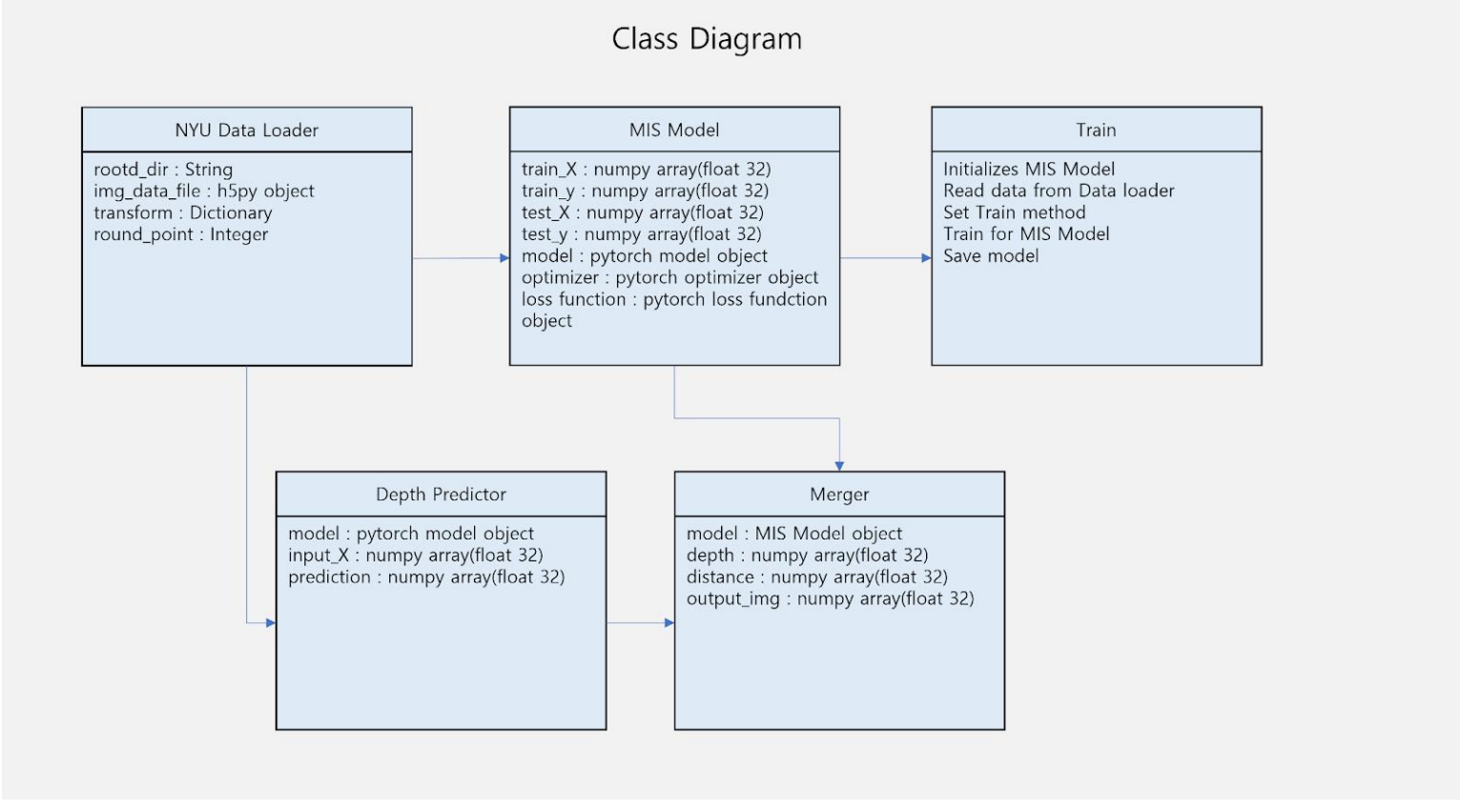


SW Architecture

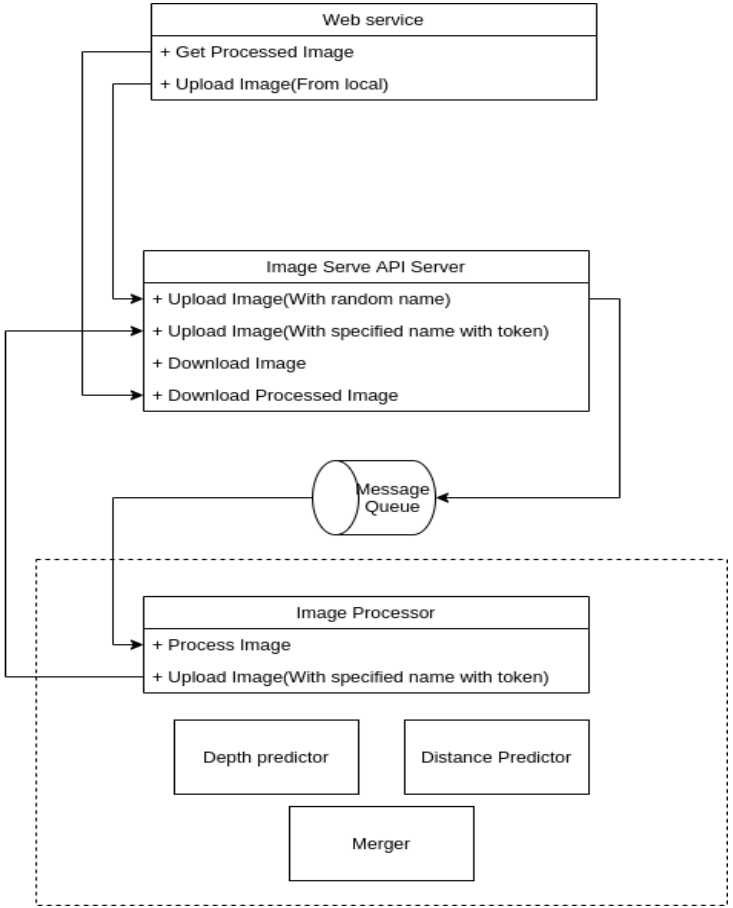




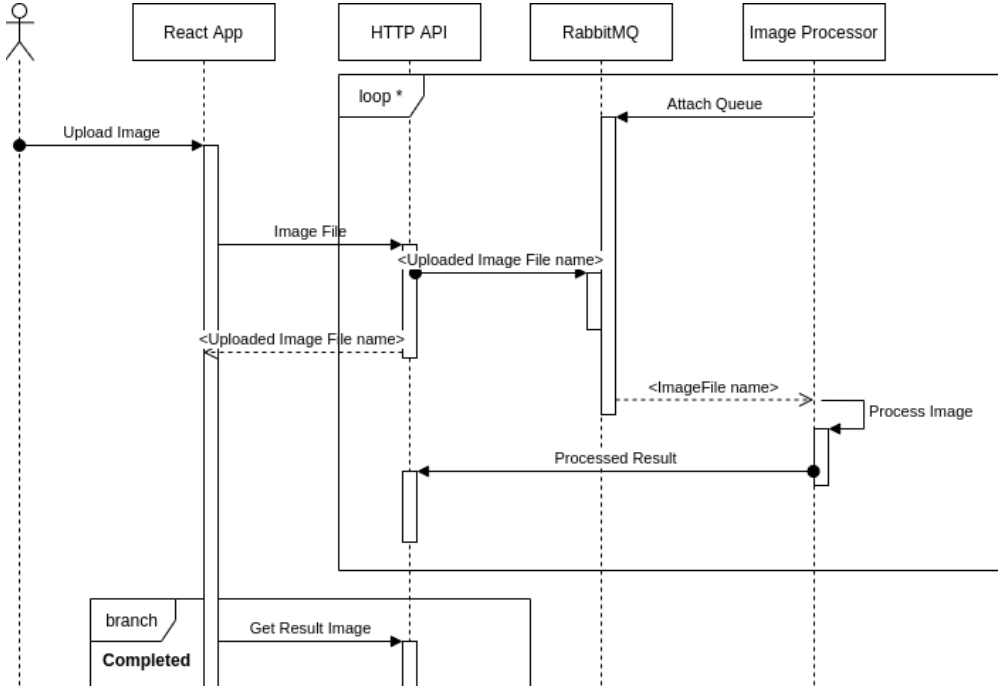
Class Diagram



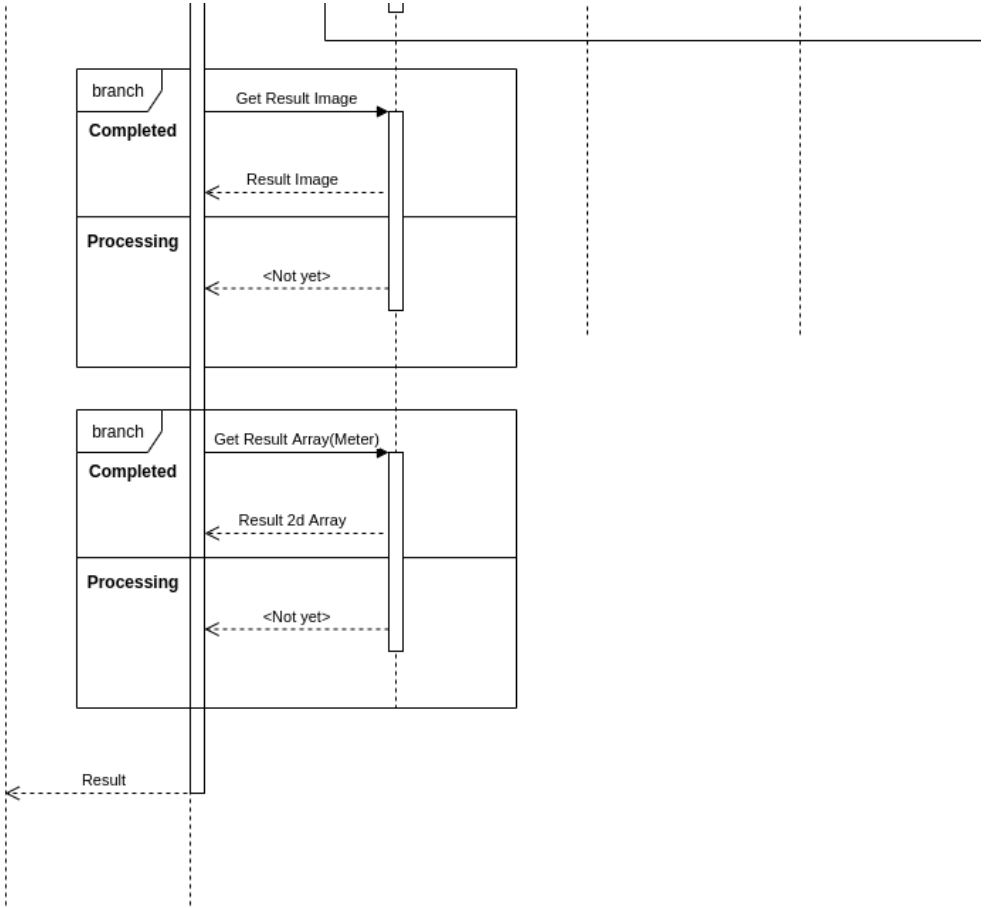
Web Service Diagram



Web Service Diagram



Web Service Diagram



5. 최종 산출물 구성 형태

Application layer

1. Python
2. HTTP 1.1/RESTful API service(Image processor)
3. Front-end Web

활용한 SW: pytorch, h5py, skelarn, fastapi, rabbitmq

Platform Layer

프로젝트에서 사용된 Application layer가 작동하는 임의의 Python Environment

Intra Layer

1. Python이 구동 가능하고 model이 gpu 메모리에 탑재가 가능한 시스템
2. docker(api service 구동)

6. Risk Analysis

현재까지 극복한 어려움들

- Distance Predictor model 개발
- Distance Predictor model에 맞는 데이터 셋의 자료형 변환, Data Loader 별도 개발
- Time cost가 비대했던 Trian 방법 개선
- **monolith 형태의 아키텍처에서 API 서빙 속도가 이미지 처리 속도에 비해 느려지게 되면 이미지 처리로 인해 API 기능이 멈출 수 있는 문제**

  MQ 기반 비동기 처리 아키텍처를 통해 워커의 scale out ability를 확보하여 이미지 처리 특성상 고질적으로 느려지는 서비스 성능을 개선

7. Success Criteria

primary use cases에 대한 Test Cases 및 테스트 방법

(생성된 모델에 대한 검증 방법)

검증 대상: NYU Depth V1

검증 방법: Predictor의 결과와 실제 distance의 RMSE 산출

검증 기준: RMSE가 4를 초과하는지 확인

품질 속성 테스트 시나리오 및 측정방법

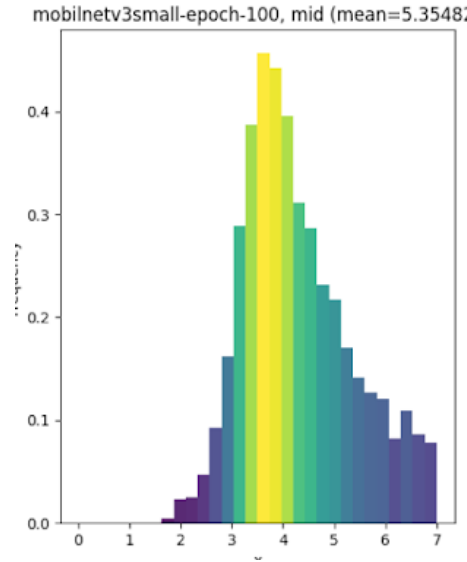
Prediction 프로그램의 RMSE 수치 = 품질 (RMSE가 낮을수록 좋은 성능)

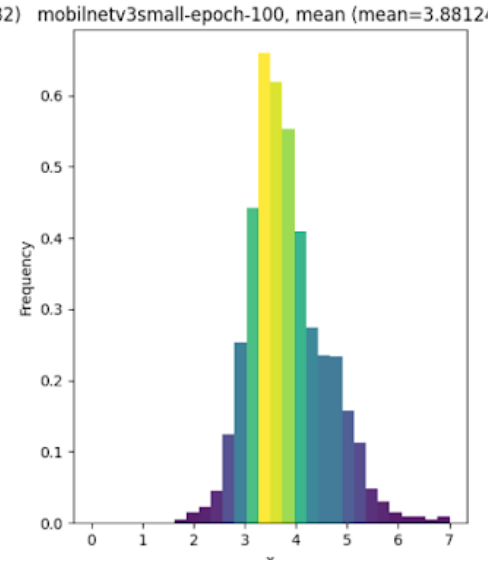
8. Result

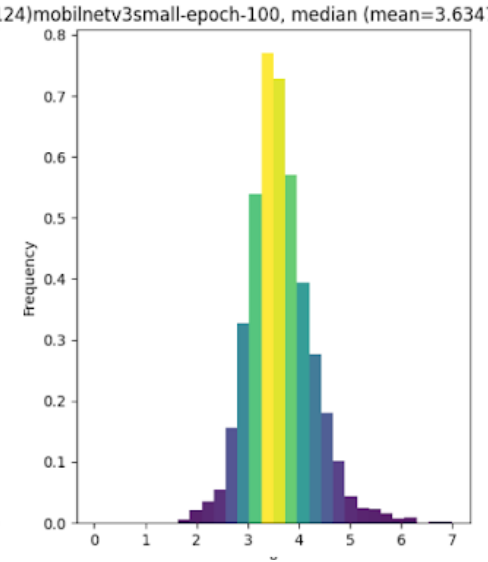
	order median	mean	median	stdev
RMSE (vgg, epoch=5)	56.5216	43.9576	41.8886	15.2391
RMSE (vgg, epoch=20)	4.45778	2.95330	2.79547	2.39000
RMSE (mobile_net_v3_small, epoch=20)	5.35786	3.88151	3.63461	2.31617
RMSE (mobile_net_v3_small, epoch=100)	5.35482	3.88124	3.63476	2.31620

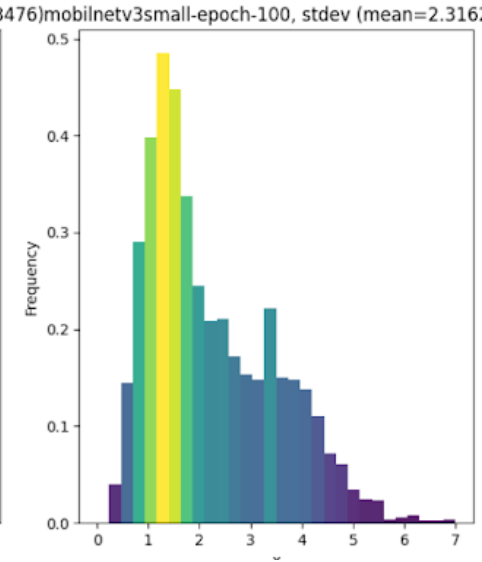
8. Result

RMSE
(mobile_net_v3_small, epoch=100)



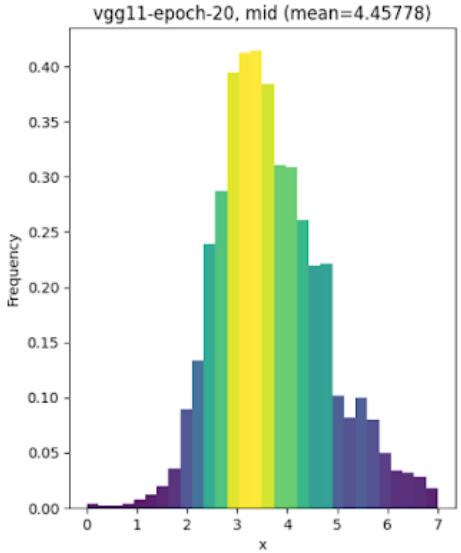


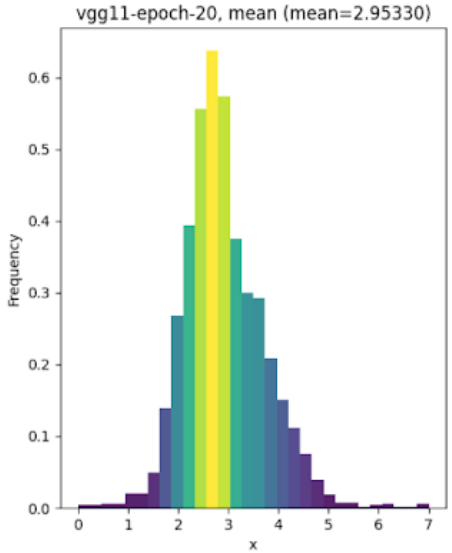


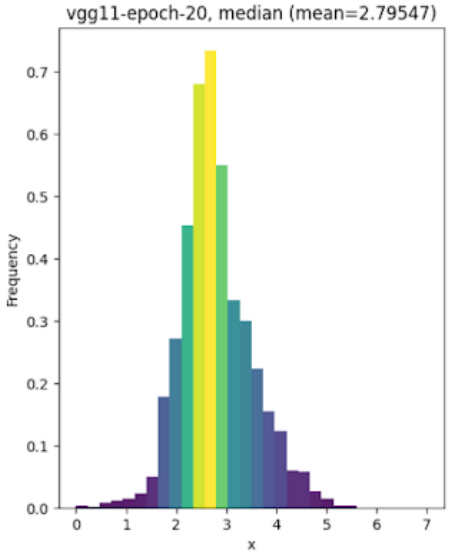


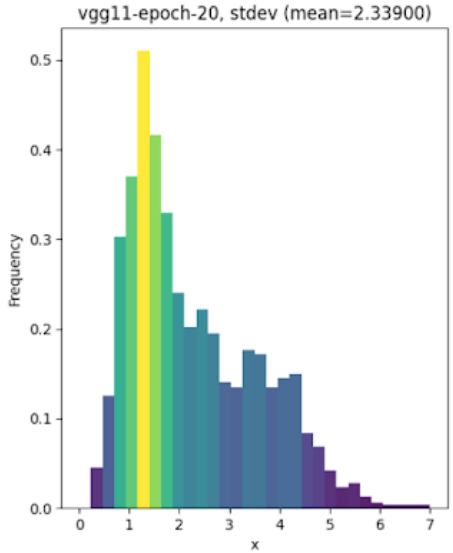
8. Result

RMSE
(vgg, epoch=20)









8. Result

Table 1: The comparisons of object-specific distance estimation with alternative approaches on the *val* subset of our constructed KITTI-object-detection-based dataset.

	Method	higher is better			lower is better			
		$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$	Abs Rel	Squa Rel	RMSE	RMSE_{log}
Car	Support Vector Regressor (SVR) [13]	0.345	0.595	0.823	1.494	47.748	18.970	1.494
	Inverse Perspective Mapping (IPM) [28]	0.701	0.898	0.954	0.497	1290.509	237.618	0.451
	Our Base Model (res50)	0.782	0.927	0.964	0.178	0.843	4.501	0.415
	Our Base Model (vgg16)	0.846	**0.947**	**0.981**	**0.150**	**0.618**	3.946	**0.204**
	Our Enhanced Model (res50)	0.796	0.924	0.958	0.188	0.843	4.134	0.256
	Our Enhanced Model (vgg16)	**0.848**	0.934	0.962	0.161	0.619	**3.580**	0.228
Pedestrian	Support Vector Regressor (SVR) [13]	0.129	0.182	0.285	1.499	34.561	21.677	1.260
	Inverse Perspective Mapping (IPM) [28]	0.688	0.907	0.957	0.340	543.223	192.177	0.348
	Our Base Model (res50)	0.649	0.896	0.966	0.247	1.315	4.166	0.335
	Our Base Model (vgg16)	0.578	0.861	0.960	0.289	1.517	4.724	0.312
	Our Enhanced Model (res50)	0.734	**0.963**	**0.988**	0.188	0.807	3.806	0.225
	Our Enhanced Model (vgg16)	**0.747**	0.958	0.987	**0.183**	**0.654**	**3.439**	**0.221**
Cyclist	Support Vector Regressor (SVR) [13]	0.226	0.393	0.701	1.251	31.605	20.544	1.206
	Inverse Perspective Mapping (IPM) [28]	0.655	0.796	0.915	0.322	9.543	19.149	0.370
	Our Base Model (res50)	0.744	0.938	0.976	0.196	1.097	4.997	0.309
	Our Base Model (vgg16)	0.740	0.942	0.979	0.193	0.912	**4.515**	0.240
	Our Enhanced Model (res50)	0.766	0.947	**0.981**	**0.173**	**0.888**	4.830	**0.225**
	Our Enhanced Model (vgg16)	**0.768**	**0.947**	0.974	0.188	0.929	4.891	0.233
Average	Support Vector Regressor (SVR) [13]	0.379	0.566	0.676	1.472	90.143	24.249	1.472
	Inverse Perspective Mapping (IPM) [28]	0.603	0.837	0.935	0.390	274.785	78.870	0.403
	Our Base Model (res50)	0.503	0.776	0.905	0.335	3.095	8.759	0.502
	Our Base Model (vgg16)	0.587	0.812	0.918	0.311	2.358	7.280	0.351
	Our Enhanced Model (res50)	0.550	0.834	**0.937**	0.271	2.363	8.166	0.336
	Our Enhanced Model (vgg16)	**0.629**	**0.856**	0.933	**0.251**	**1.844**	**6.870**	**0.314**

	order median	mean	median	stdev
RMSE (vgg, epoch=20)	4.46	2.95	2.80	2.39

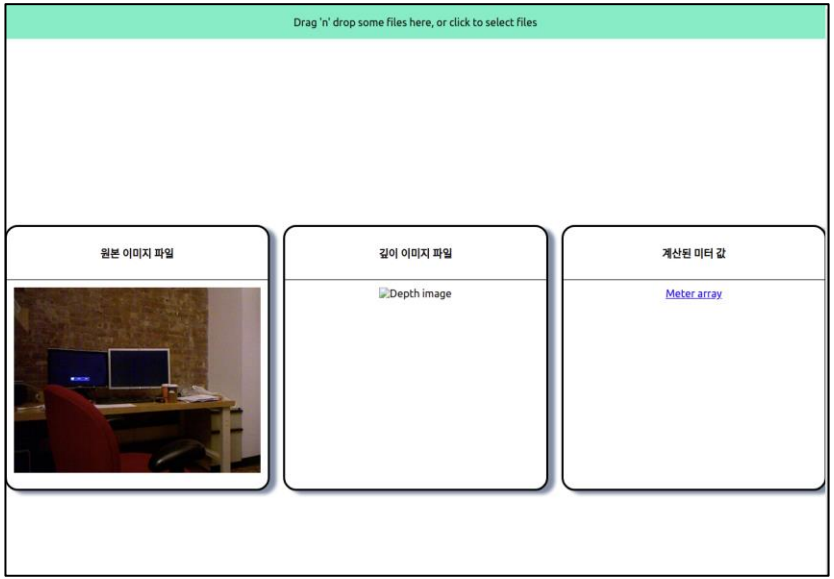
Learning Object-Specific Distance From a Monocular Image (2019)

아쉬웠던 점

한정된 자원과 시간

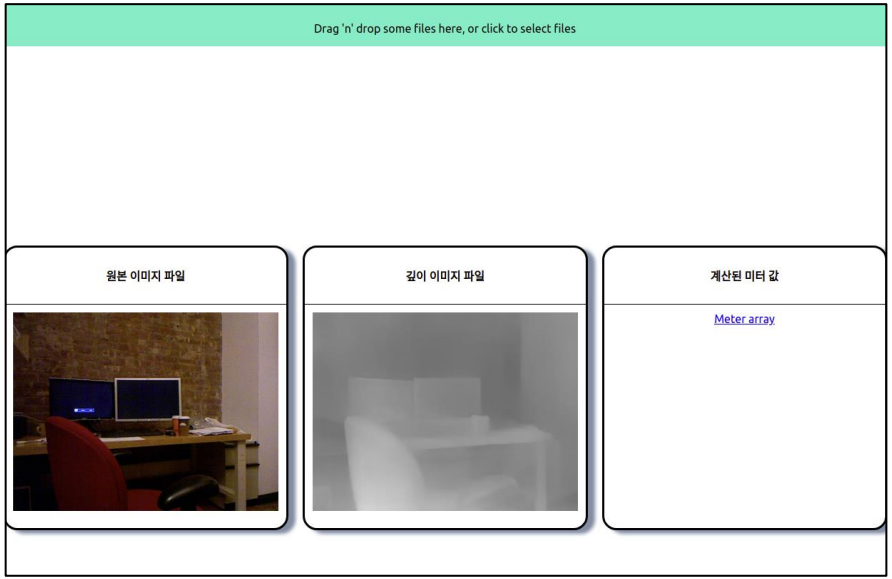
8. Result

Front-end Web



8. Result

Front-end Web



예상되었던 문제점

대형 서버가 아닌 개인 데스크탑 환경(AMD 3600, RTX 3070 8gb, 16gb memory)에 서버 구축 → 동시에 최대 1명의 유저만 predictor 사용 가능

↓

- MQ 기반의 비동기 서비스 아키텍쳐로 워커의 Scale out 가능
- 따라서 상대적으로 느린 이미지 처리 과정의 문제를 해결

예상되는 문제점

NYU Depth V1은 실내 사진 데이터

실외 사진에 대해서는 정확도를 보장하기 힘들다.

10. 진행 스케줄

	3월 1st half	3월 2nd half	4월 1st half	4월 2nd half	5월 1st half	5월 2nd half	6월 1st half
프로젝트 구상							
논문 리뷰							
기존 연구 코드 분석 및 개선							
Risk Analysis							
딥러닝 모델 개발							
모델 검증							
모델 개선							
데모 제작							

11. List of Reference

MegaDepth: Learning Single-View Depth Prediction from Internet Photos

Cornell University/Cornell Tech (2018)

https://www.cs.cornell.edu/projects/megadepth/

Lesrning Object-Specific Distance From a Monocular Image

Cornell University (2019)

https://openaccess.thecvf.com/content_ICCV_2019/papers/Zhu_Learning_Object-Specific_Distance_From_a_Monocular_Image_ICCV_2019_paper.pdf

Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks

https://arxiv.org/pdf/1506.01497.pdf

12. List of Reference

참고 OSS(Git)

https://github.com/zhengqili/MegaDepth

https://github.com/herbwood/pytorch_faster_r_cnn