[피어세션]

목차

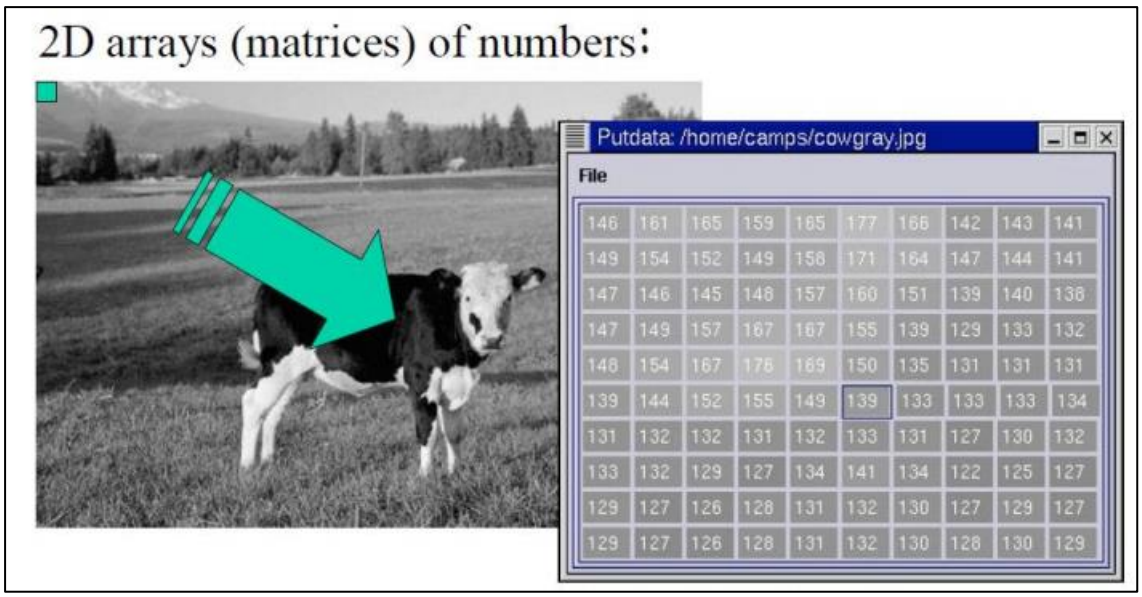
1. 면접과 느낀점
2. 이미지 데이터에 대하여
3. Benchmark Dataset in Image
4. Panorama Image
5. Mesh Modeling
6. TSM (Traffic Sign Mapping)
7. 추후 소개드릴 내용 (이미지 처리를 위한 라이브러리, SLAM 알고리즘)

[1. 면접과 느낀점]

- 직무: AI/Big Data (설비기술)
- 실제로 컴퓨터공학 + AI를 전공한 지원자는 소수 (기계공 8 : 컴공 2)
- 신입사원에서 바라는 것은 뛰어난 기술 스택이 아니라 인성 & 분석 능력
- 면접은 한번 떨어져봐야 무엇이 부족한지 느낌 (넥타이, 옷차림, 당황할 때의 모습 등)
- 지원자들끼리 친해져서 정보를 공유하는 것도 좋은 방법
- (질문) 자신의 단점?

[2. 이미지 데이터에 대하여]

- Gray Scale: 1 channel, Color: 3 channels, Depth/Infra: 4 channels

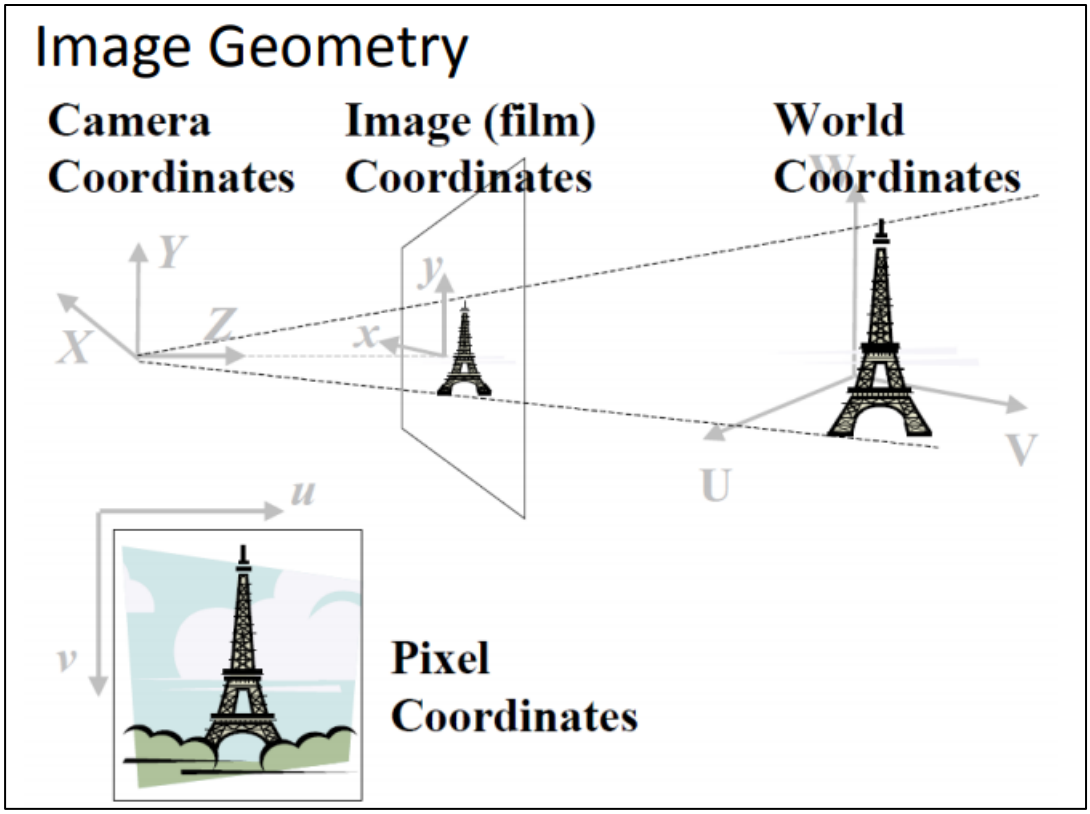


- Pixel Value Range: 0-255 (SDR <-> HDR)

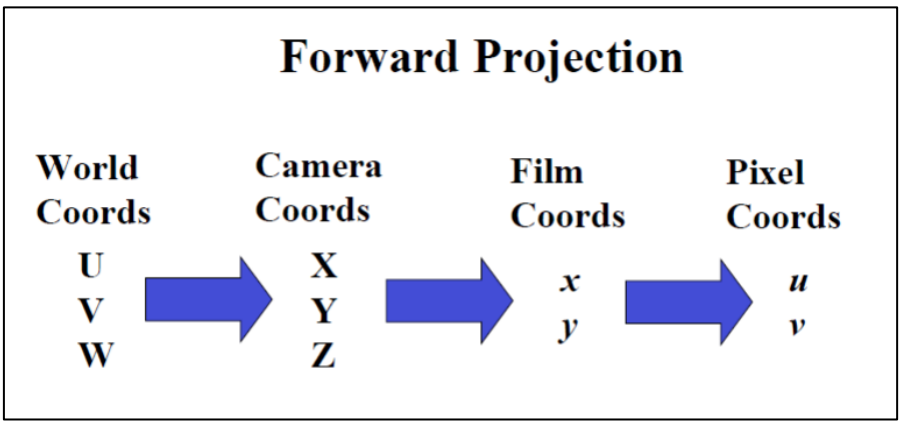
- 인공지능의 효율적인 학습/테스트를 위해서는 Image Normalization 수행

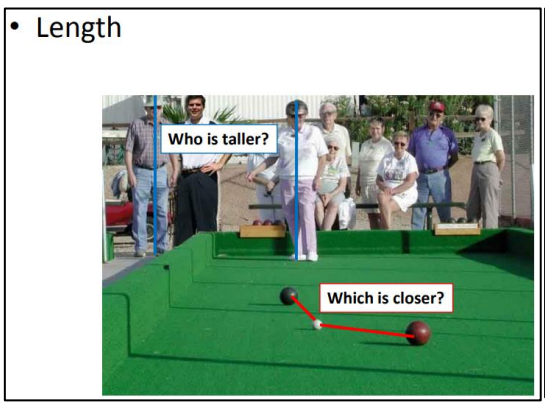
```
def test(self, image):
  image = image / 255.0
  label = self.model.predict(image)
  label = np.argmax(label, axis=1)
  print("Prediction for Images:", label)
  return label
```

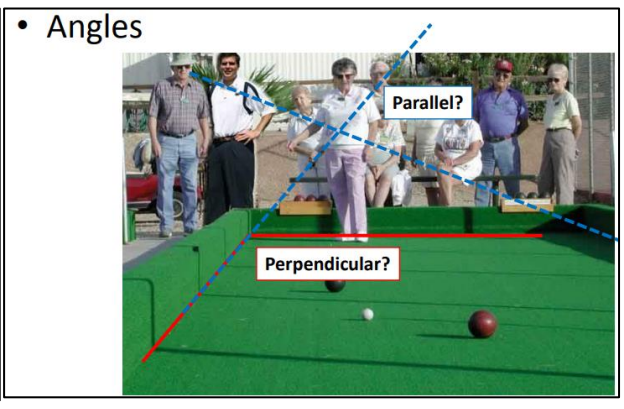
- Projection과 Reconstruction의 과정을 위한 좌표계 변환 프로세스



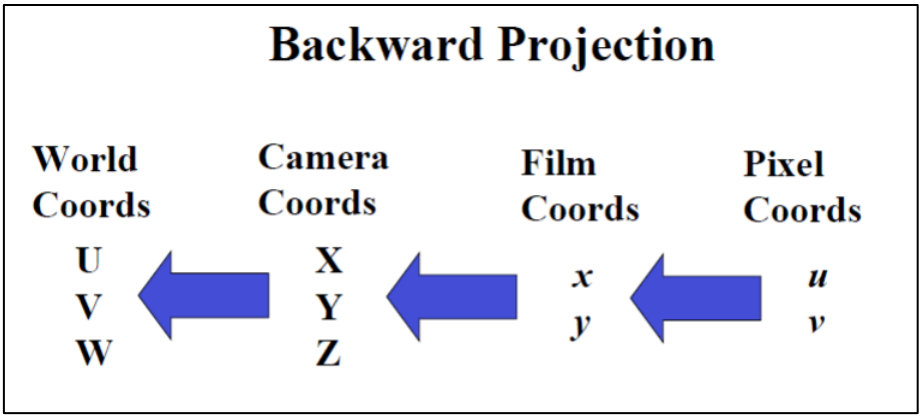
- 3D -> 2D (Projection)

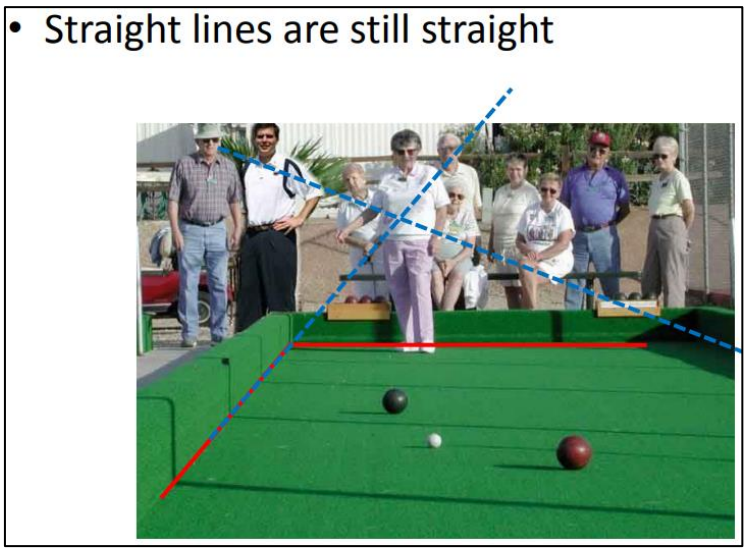






- 2D -> 3D (Reconstruction)

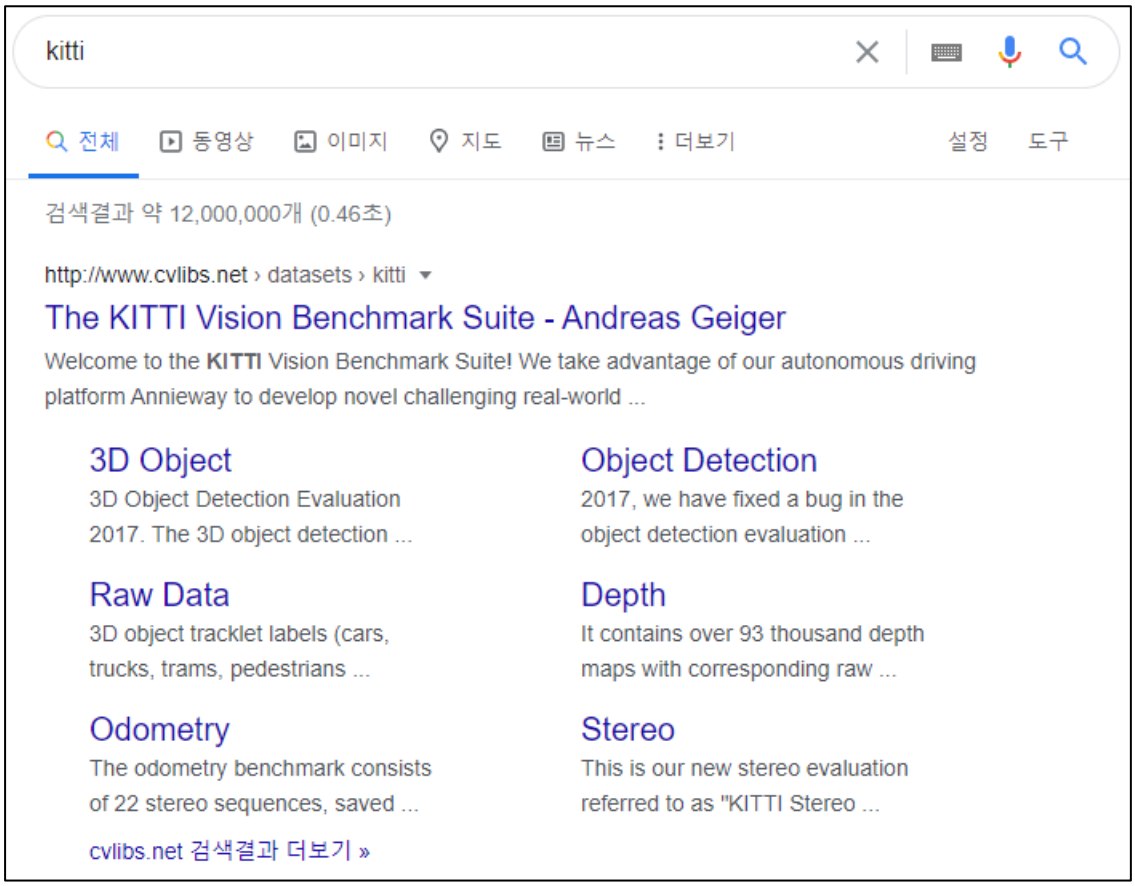




[3. Benchmark Dataset in Image]

- 일반적인 Image Classification: MNIST, CIFAR와 같은 숫자/풍경/동물 제공

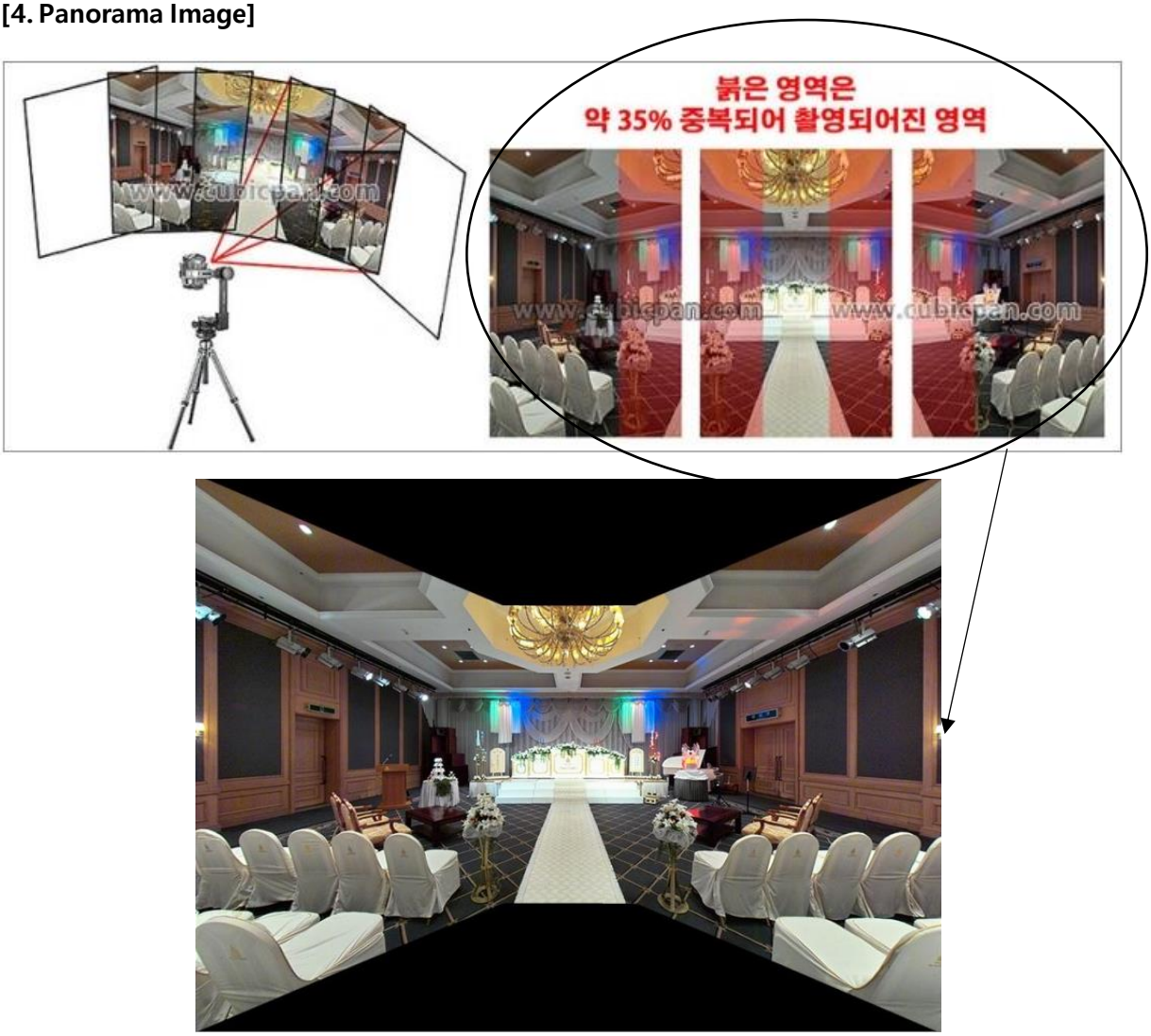
- 자율주행을 위한 Dataset: KITTI, Waymo와 같은 도로상황을 중점적으로 제공



URL(KITTI): http://www.cvlibs.net/datasets/kitti/

URL(CVPR): https://rdx-live.tistory.com/90

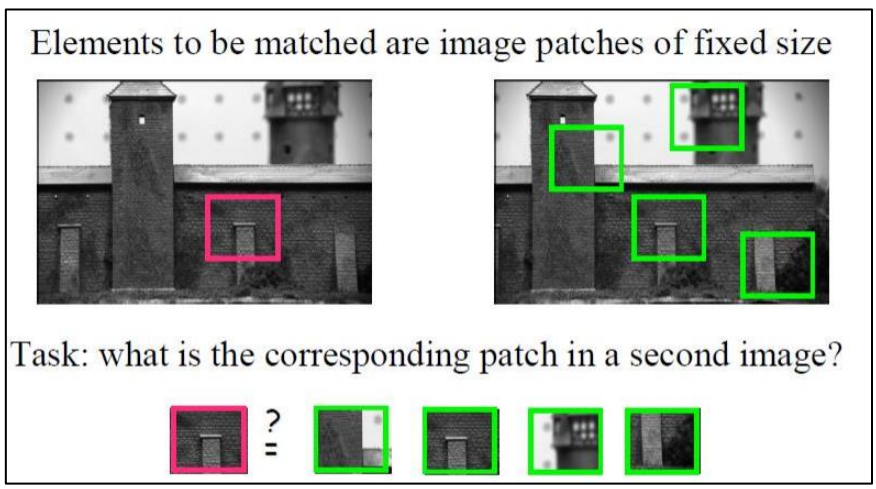
[4. Panorama Image]



- 이미지 간의 중복이 필요한 이유: Matching

- Template Matching (==Exhaustive Search)

URL: 2003 Template matching for detecting hands in cluttter - YouTube



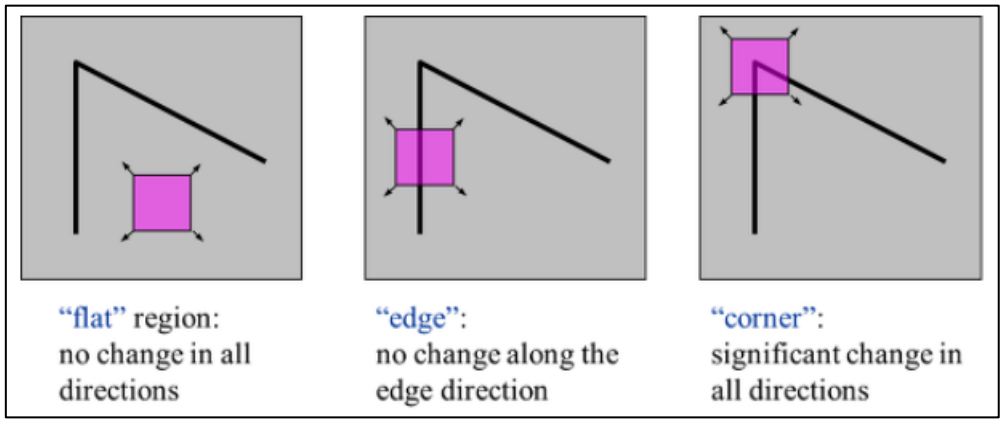
- Key Point Matching

영상 특징점, keypoint, interesting point

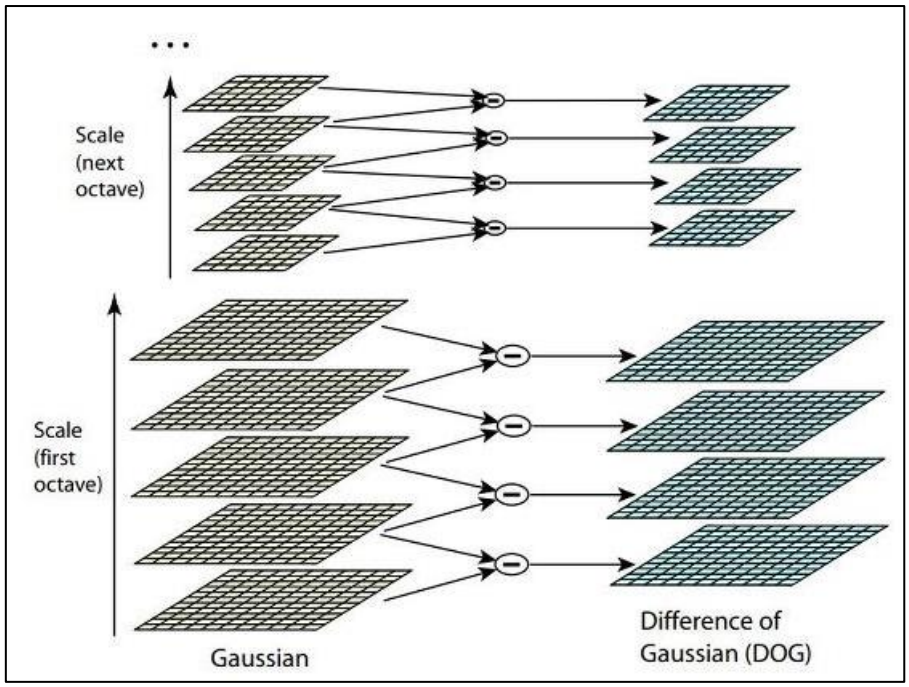
조건1: 물체의 형태나 크기, 위치가 변해도 쉽게 식별 가능

조건2: 카메라의 시점, 조명이 변해도 영상에서 해당 지점을 쉽게 탐색 가능

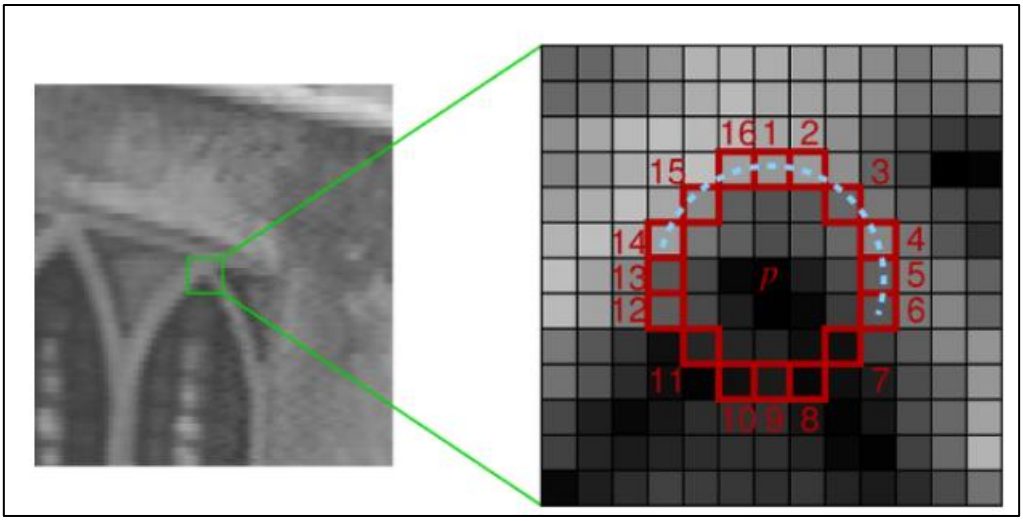
Harris Corner[1998]



SIFT[2004]



FAST[2006]




```python
def detectAndDescribe(self, image):
  # detect key points and features using SIFT
  gray = cv.cvtColor(image, cv.COLOR_BGR2GRAY) # rgb -> gray conversion
  descriptor = cv.SIFT_create()
  (kps, features) = descriptor.detectAndCompute(image, None)
  kps = np.float32([kp.pt for kp in kps]) # convert kp object to numpy
  return (kps, features)

def matchKeypoints(self, kps1, kps2, features1, features2, ratio, reprojThresh):
  # image mathcing & find homography matrix using bruteforce method
  matcher = cv.DescriptorMatcher_create("BruteForce") # exhaustive search
  rawMatches = matcher.knnMatch(features1, features2, 2) # return top two matches
  matches = []
  for m in rawMatches:
    if len(m) == 2 and m[0].distance < m[1].distance * ratio:
      # apply David Lowe's ratio test for false-positive match pruning
      matches.append((m[0].trainIdx, m[0].queryIdx))
  if len(matches) > 4:
    # need at least 4 matches for homography matrix calc
    pts1 = np.float32([kps1[i] for (_, i) in matches])
    pts2 = np.float32([kps2[i] for (i, _) in matches])
    (H, status) = cv.findHomography(pts1, pts2, cv.RANSAC, reprojThresh)
    return (matches, H, status)
  return None
```

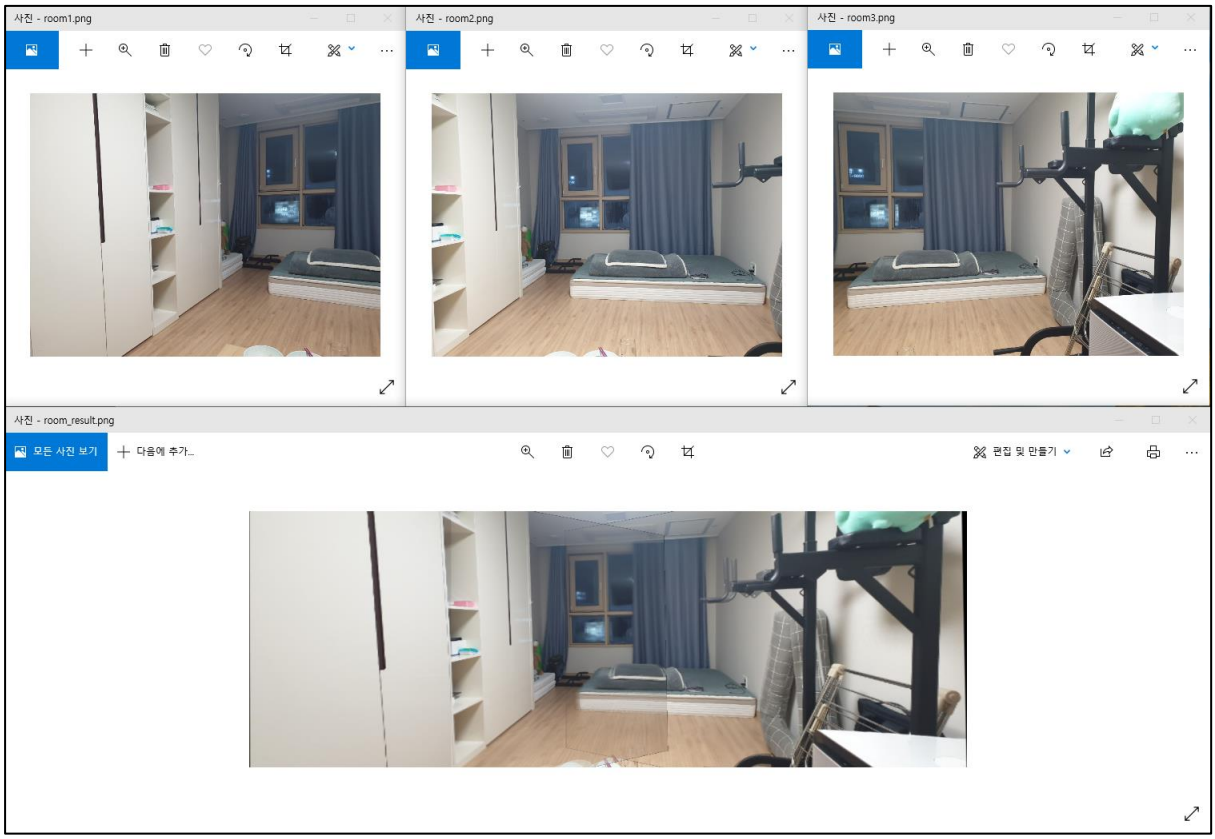
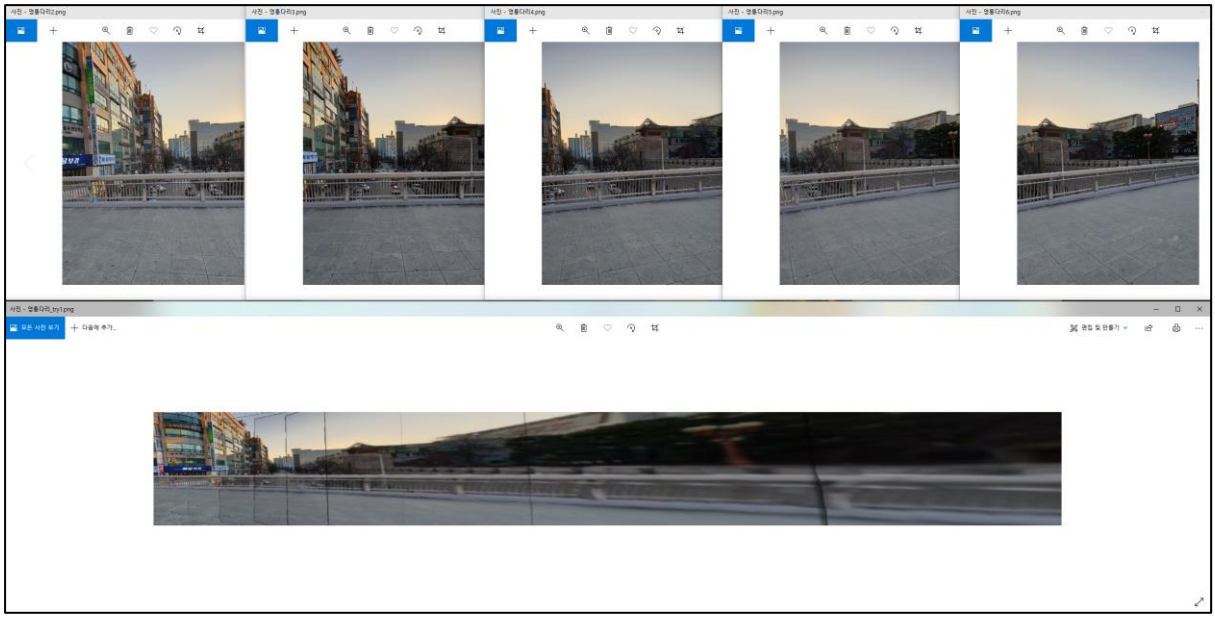
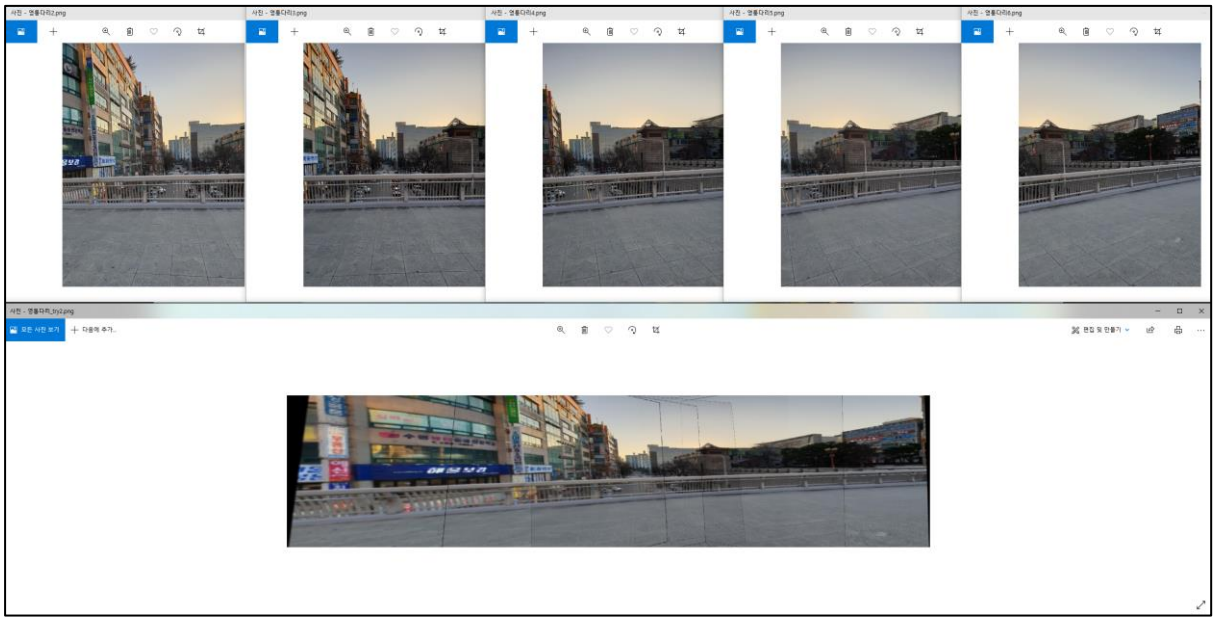


사진 - room1.png
사진 - room2.png
사진 - room3.png
사진 - room_result.png
모든 사진 보기
다음에 추가...
편집 및 만들기





- 이미 구현되어 있는 라이브러리 클래스의 성능이 훨씬 좋음 (Stitcher Class) -> 분석 & 활용
- Document에는 제공되지 않은 내용 (다수의 사진 적용, Matching 시 중간 이미지, 경계 처리)
- Google Colab 환경에서 Juypter Notebook 코딩 -> 모듈화 & 패키지화 해서 Github 업로드

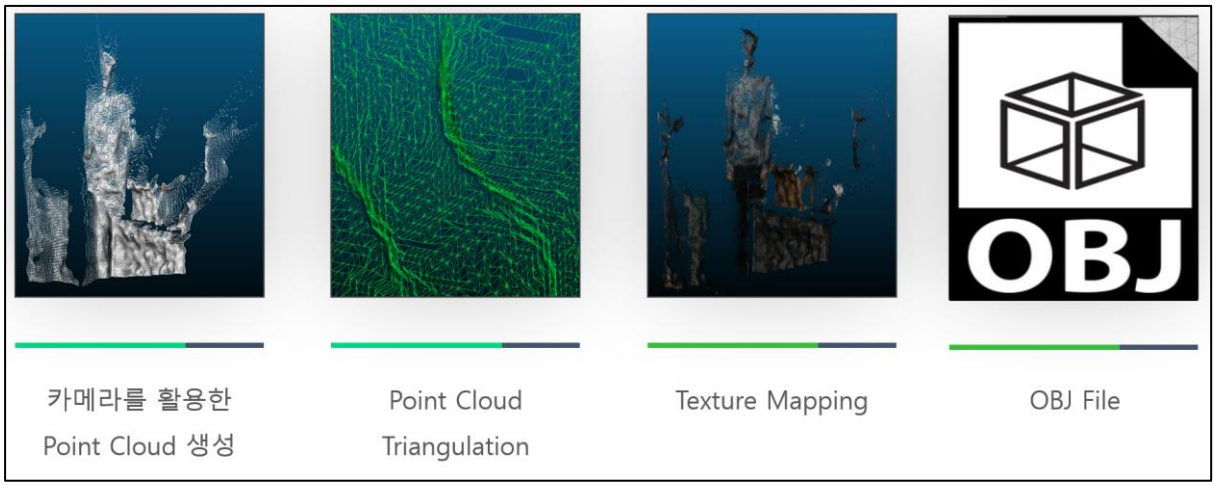
[5. Mesh Modeling]

- 2차원 이미지로는 정보의 표현력이 제한
- 네이버, 지리정보원 등 2차원 지도를 3차원으로 확장하려는 움직임 (Camera -> LiDAR)
- 프로젝트의 의의

  일반인이 3차원 데이터를 생성할 때 LiDAR 센서는 사용 불가능

  범용적인 3D Modeling 기술을 위한 파이프라인 구축

  기존 프로그램을 통해서만 생성한 Mesh 데이터를 코드로 생성할 수 있도록 모듈화



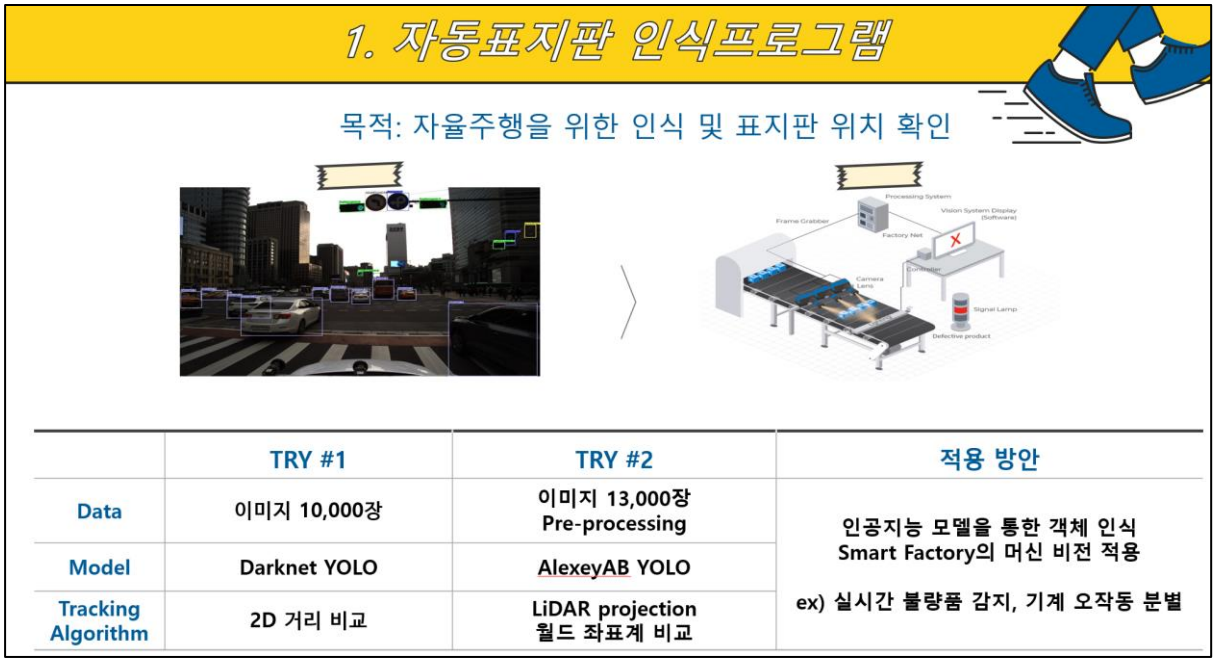
- (동영상 시연)

Raw Depth Image를 통해 생성한 3차원 데이터는 전체적으로 Noisy

Multi-Frame Normalization을 통해 생성한 3차원 데이터는 Boundary 주변만 Noisy

Deep Learning을 적용한 3차원 데이터는 일부 필요한 정보를 잃어도 정확도 향상

[6. TSM (Traffic Sign Mapping)]



	TRY #1	TRY #2	적용 방안
Data	이미지 10,000장	이미지 13,000장 Pre-processing	인공지능 모델을 통한 객체 인식 Smart Factory의 머신 비전 적용 ex) 실시간 불량품 감지, 기계 오작동 분별
Model	Darknet YOLO	AlexeyAB YOLO	
Tracking Algorithm	2D 거리 비교	LiDAR projection 월드 좌표계 비교	

- 프로젝트의 의의

  Benchmark 데이터셋 (해외 도로상황)이 아닌 직접 수집한 국내 데이터셋 활용

  실시간 처리를 위한 YOLO 모델 사용

  ↙ Tracking 알고리즘의 개선으로 정확도 향상

Backbone	GPU	FPS
ResNet-101	Titan X	53
ResNet-152	Titan X	37
Darknet-53 (paper)	Titan X	76
Darknet-53 (this impl.)	1080ti	74

[7. 추후 소개드릴 내용 (이미지 처리를 위한 라이브러리, SLAM 알고리즘)]

- 실질적인 이미지 프로세싱은 대부분 라이브러리 활용 (CV domain 때 필요할 듯)

- SLAM == Simultaneous Localization And Mapping (in 로보틱스)