2023.08.06

2023년 하계 UST 연구인턴십

결과보고서

UST-KITECH School Robotics Group
지도교수 : 고광은
인턴학생 : 박민건

Contents

1 건고추 이물 검출

2 오리 방혈 자동화

3 분광데이터를 이용한 고추가루 맵기 분류

4 UST 하계 인턴십 소감

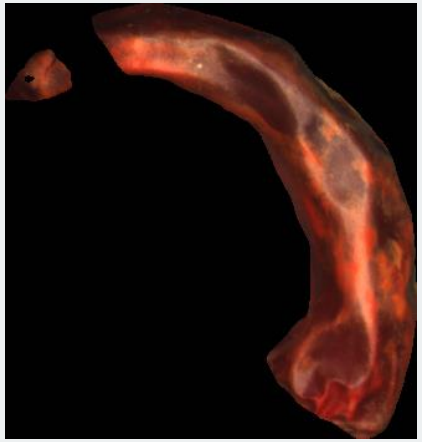
1. 건고추 이물 검출

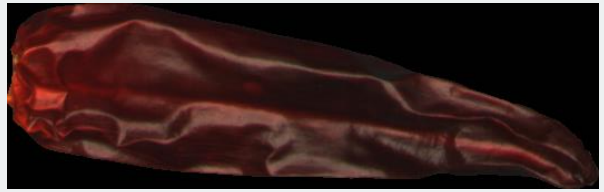
진행 기간 : 23.07.03 ~ 23.07.12

추진 배경

- 기존의 건고추 분류는 인적 자원의 반복 작업으로 이뤄지나, 이것을 이미지 처리 기술을 통해 자동화 하는 것이 목표.

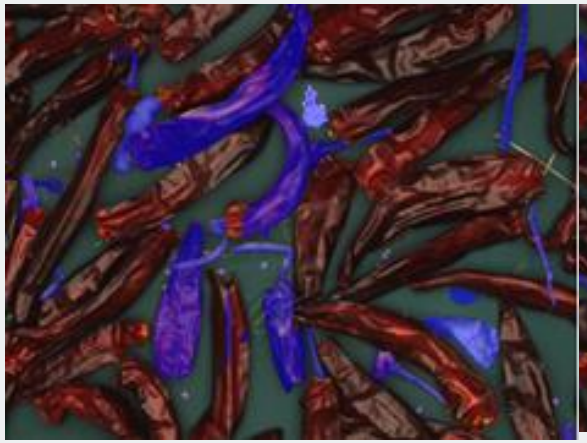
건고추 이물 검출은 건고추 분류 자동화의 일부

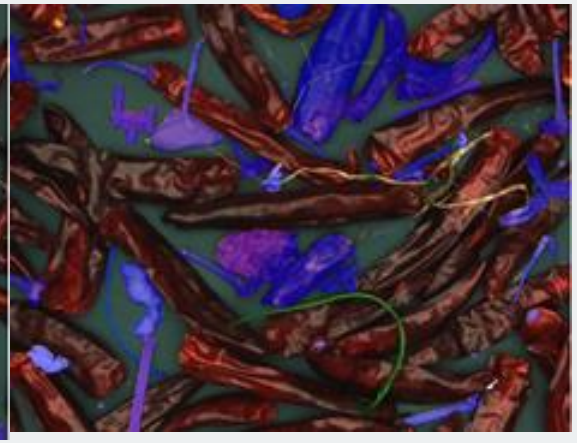




수행 결과

- **실을 제외한 대부분의 건고추 이물을 검출하는 데 성공.**
- **실도 검출할 수 있게 미세 조정 진행 중**





1. 건고추 이물 검출

수행 역할

- 건고추 라벨링 작업 및 전처리 보조
- 라벨링을 빠르고 정확하게 진행하기 위한 라벨링 프로그램 개발(콘솔)

<코드1. 건고추 분류 라벨러 프로그램>

```
import shutil

BASE_PATH = "C:\labeling\mask_data_h\pep_"
num = "0062"
dir_path = BASE_PATH+num
folder_dir = [dir_path+'/0', dir_path+'/1']

for item in folder_dir:
    if not os.path.exists(item):
        os.makedirs(item)   000?"g414

cv2.namedWindow("Image", cv2.WINDOW_AUTOSIZE)
cv2.setWindowProperty("Image", cv2.WND_PROP_FULLSCREEN, cv2.WINDOW_FULLSCREEN)    # maximize the window
cv2.moveWindow("Image", 50, 50)

for file_name in file_list:
    if file_name.endswith(".png"):
        image = cv2.imread(os.path.join(dir_path, file_name))

        cv2.imshow("Image", image)

        while True:
            cv2.imshow("Image", image)

            # Wait for key event
            key_event = cv2.waitKeyEx(0)

            # Check the key event type
            if key_event == ord('0'):
                shutil.move(os.path.join(dir_path, file_name), os.path.join(dir_path+"/0/", file_name))
                break
            elif key_event == ord('1'):
                shutil.move(os.path.join(dir_path, file_name), os.path.join(dir_path+"/1/", file_name))
                break
            elif key_event == 27:  # ESC key
                exit()
            else:    # ESC key
                break

cv2.destroyAllWindows()
```

2. 오리 방혈 자동화

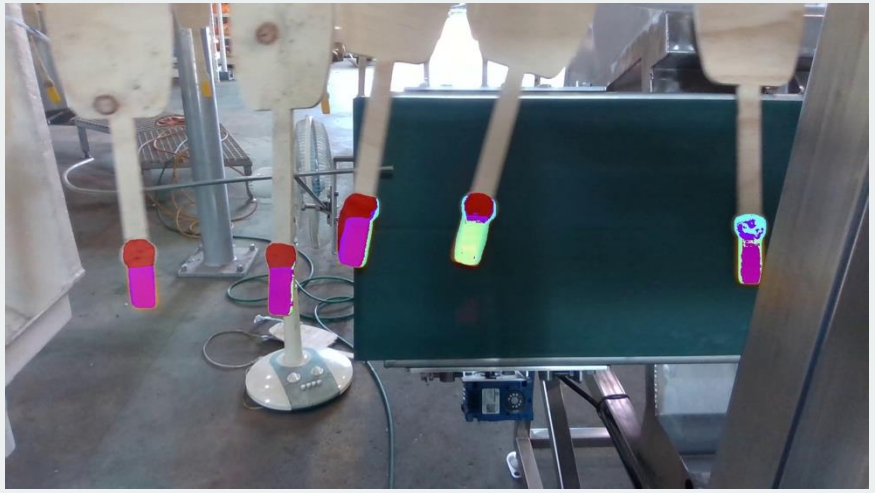
진행 기간 : 23.07.13 ~ 23.07.21

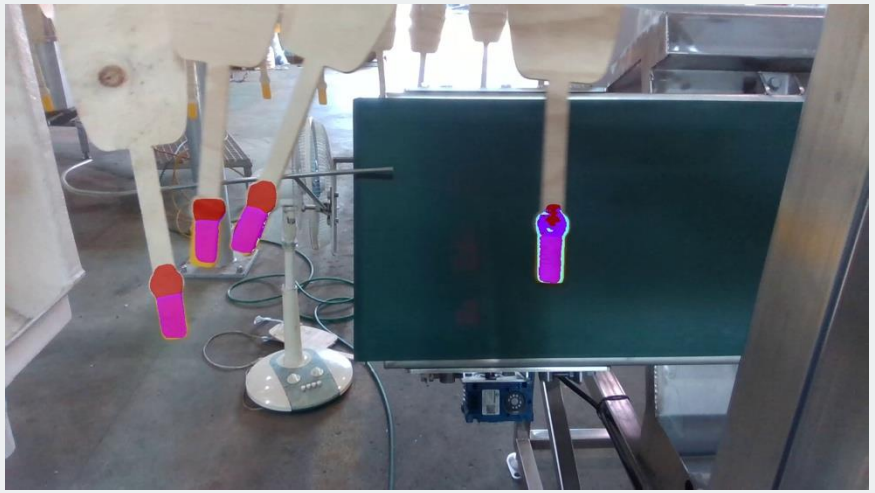
추진 배경

- 오리를 상품화하는 과정 중 오리를 오리의 피를 방출하는 과정이 있음.
- 오리의 방혈은 현재는 인적 자원에 의해 이뤄지고있으나 이는 정신적으로 신체적으로 힘든 작업. 이것을 자동화 솔루션을 제공하여 해결하고자함.

수행 결과

- 오리가 레일을 따라 이동하고, 이를 카메라와 이미지 인식 모델로 따라가며 머리를 인식함.

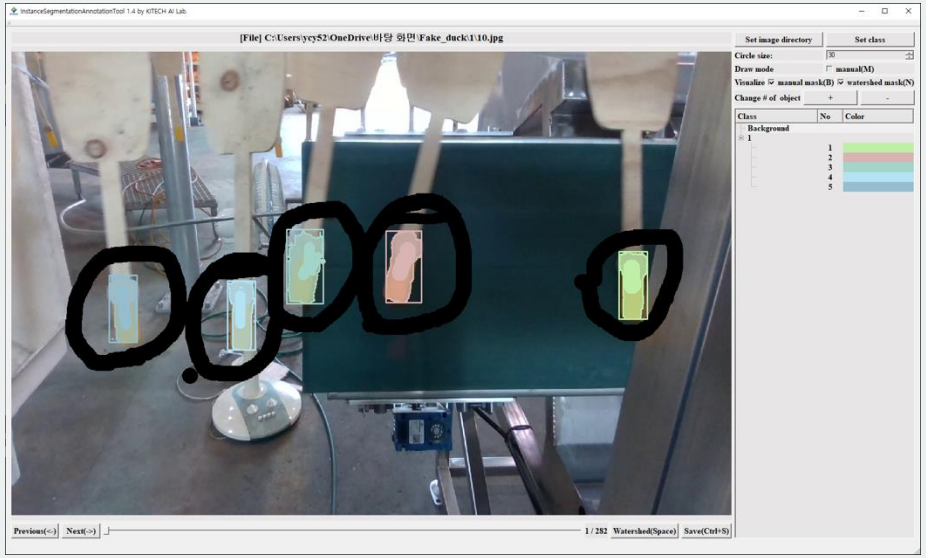


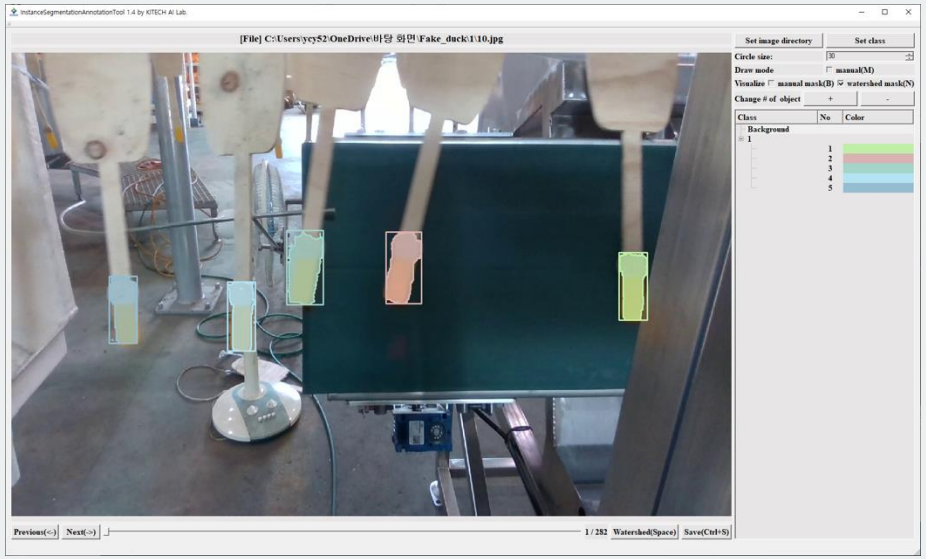


2. 오리 방혈 자동화

수행 역할

- 오리 이미지에서 머리부분을 Detecting하기 위해 라벨링 작업 진행
- 282장의 Fake 오리 사진의 머리 부분을 라벨링





3. 분광데이터를 이용한 고추가루 맵기 분류(Classification)

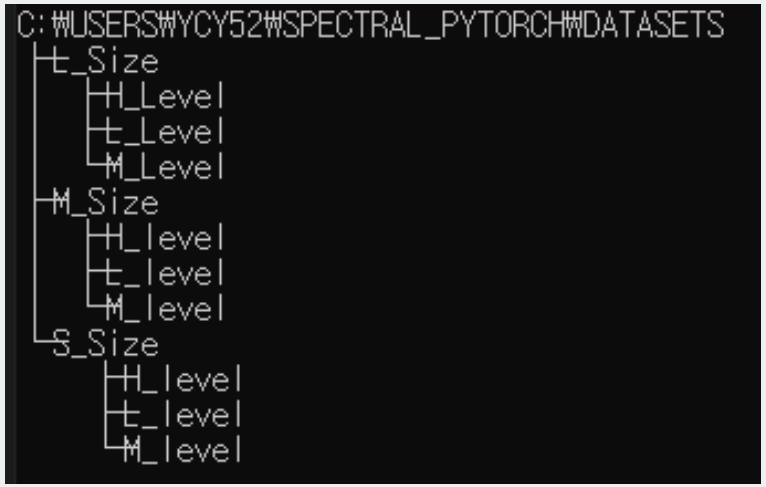
진행 기간 : 23.07.21 ~ 23.08.07

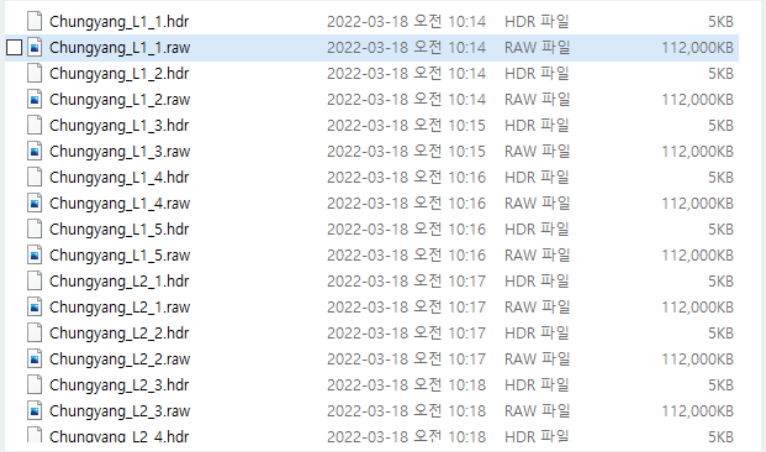
추진 배경

- 고추가루는 현재 인적자원의 동원없이는 맵기를 분류 불가.
- 256채널의 분광 데이터를 이용하여 특징을 찾고, 이를 분류하여 자동화하고자함.

수행 결과

- 기존의 hyper spectral CNN 분류 모델을 기반으로 라벨 클래스를 추가하고 전처리 작업을 수정하여 학습을 수행. 현재 모델의 구조 수정이 필요하여 수치적 결과는 내지못함.





3. 분광데이터를 이용한 고추가루 맵기 분류(Classification)

수행 역할

- 분광 데이터를 불러와서 [.raw, .hdr, label] 을 반환하는 PathCombiner Class 생성 및 작성
- 모델 내 전처리 파트 수정
- 전체적인 모델

```
class PathCombiner:

    def __init__(self, dataset_path):
        '''
            dataset_path: Project Root Path
        '''
        self.data_path = dataset_path
        self.current_path = os.getcwd()

    def get_folders(self, path):
        return [os.path.join(path, folder) for folder in os.listdir(path) if os.path.isdir(os.path.join(path, folder))]

    def get_files(self, path, ext):
        return [os.path.join(path, file) for file in os.listdir(path) if os.path.isfile(os.path.join(path, file)) and file.endswith(ext)]
```

3. 분광데이터를 이용한 고추가루 맵기 분류(Classification)

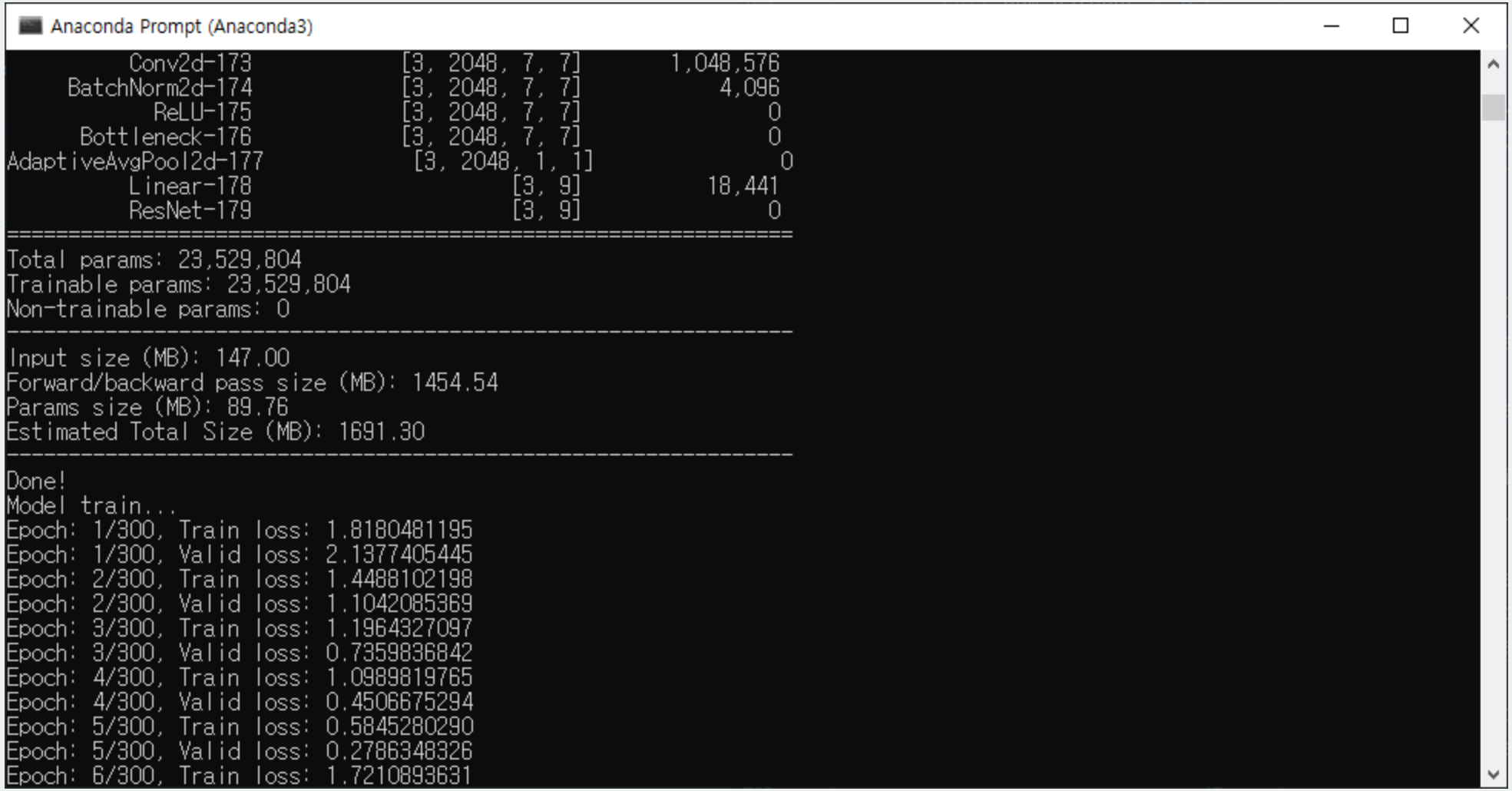
```
class PathCombiner:
        def get_level_file_paths(self):
            data_path = os.path.join(self.current_path, self.data_path)
            if not os.path.exists(data_path):
                print(f"{data_path} 경로가 존재하지 않습니다.")
                return

            hdr_files = []
            raw_files = []
            labels = []
            level_index = 0

            size_folders = self.get_folders(data_path)

                for size_index, size_folder in enumerate(size_folders):
                    level_folders = self.get_folders(size_folder)

                    for level_index_in_size, level_folder in enumerate(level_folders):
                        hdrs = self.get_files(level_folder, '.hdr')
                        raws = self.get_files(level_folder, '.raw')

                        hdr_files.extend(hdrs)
                        raw_files.extend(raws)

                        label = size_index * 3 + level_index_in_size
                        labels.extend([label] * len(hdrs))

                    level_index += 1

                # print(label)
                return hdr_files, raw_files, labels`
```

3. 분광데이터를 이용한 고추가루 맵기 분류(Classification)

```
class CustomGenerator(Dataset):
    def __init__(self, dataset_path, class_num, is_train):
        self.data_path = dataset_path
        self.cls_num = class_num
        self.is_train = is_train
        self.data_group, self.label_group = self.load_data(self.data_path, self.cls_num, self.is_train)

    def __len__(self):
        return len(self.data_group)

    def __getitem__(self, item):
        input = torch.FloatTensor(self.data_group[item])
        target = torch.tensor(int(self.label_group[item][0]))

        return input, target

    def load_data(self, dataset_path, class_num, is_train):
        data  []
        path_combiner = PathCombiner("datasets")
        hdr_files, raw_files, labels = path_combiner.get_level_file_paths()

        for hdr, raw in zip(hdr_files, raw_files):
            hr_open = envi.open(hdr, raw)
            hr_open = np.array(hr_open.load())
            hr_open = self.preprocess(hr_open)
            data.append([hr_open])  # single-channel 경우 (1, h, w)로 맞춰줌

          labels = np.array(labels, dtype=np.int32)
        total_labels = np.reshape(labels, (-1, 1))  # reshape: (1, 210) -> (210, 1)
          return np.array(data), total_labels  # data shape : (210, 3, 16, 1376), label shape : (210, 1)
```

```
def preprocess(self, data):
        h, w, c = data.shape
        crop_data = data[int(h/2)-60:int(h/2)+4, int(w/2)-60:int(w/2)+4]
        for i in range(0, c):
            c_data = crop_data[:, :, i]

            if i == 0:
                preprocess_data = c_data
            else:
                preprocess_data = np.concatenate((preprocess_data, c_data), axis=1)

        return preprocess_data
```

3. 분광데이터를 이용한 고추가루 맵기 분류(Classification)



3. UST 하계 인턴십 소감

"연구자에서 개발자로"

저는 개발자 성향이 강한 학생입니다. 대학교 3학년때부터 풀스택 외주 개발자로 일하였고, 이런 자신의 커리어에 대해 자신이있었습니다. 또 후회없이 열심히 살았다고 생각했습니다. 그런데, 대학원으로 진로를 변경한 후 기존에 준비해오던 모든 것들이 진학에 크게 도움이 되지 않는 이런 새로운 필드에서 자신, 연구자라는 새로운 직업에 대해 진정성, 절박함이 부족하지 않았는지에 대한 생각을 이번 이넌십으로 하게 되었습니다.

생산과학기술원 로봇그룹에 와서 자신의 분야에 열심히, 또 열정적으로 살아가며 생기있는 눈으로 논쟁하는 대학원 선배님들, 박사님들을 보고 저 자신이 이 분야에 대해 얼마나 관심이 있는지, 또 대학원에 대해 다시 한번 깊게 통찰하고, 마음을 다질 수 있는 좋은 시간이였습니다.