Real World Problem Project Using OpenCV

AVGenius

AVGenius

Midterm Project

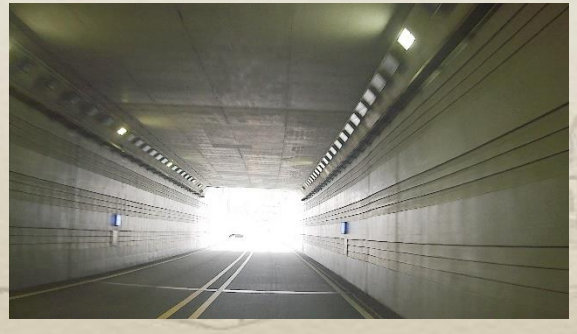
- 프로젝트 개요
  - 주제 : 악천후 환경에서 Camera Fail-Safe를 위한 Camera 영상의 필터링 방법
  - 악천후 환경에서 자율주행차 주행 중 객체 인식(Object Detection)에 주로 활용하는 Camera 센서의 오염이나 급격한 환경 변화로 인해 객체 인식에 오류가 발생할 가능성이 있는 경우 해당 영상의 이미지에 필터링(filtering)을 적용하여 해결
  - 악천후 환경 유형

야간(night)

강우(rain)

안개(fog)

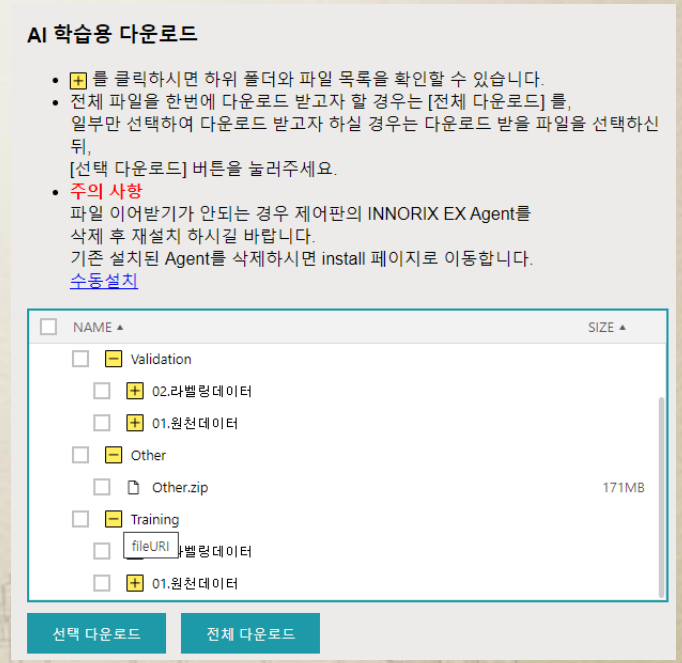
역광(backlight)



급격한 조도 변화(sudden change in illumination)

Midterm Project

- 프로젝트 진행
  - 악천후 환경 주행 데이터 취득
    - AI-Hub(https://www.aihub.or.kr/) 공공 데이터셋의 딥러닝용 악천후 환경 자율주행 데이터셋 활용(2023년 상용 자율주행차 악천후 데이터)
    - 도심로의 야간, 안개, 강우, 역광, 급격한 조도 변화 환경에 해당하는 데이터 선별



Midterm Project

- 프로젝트 진행
  - 악천후 환경 별 필터링 적용
    - 야간 환경의 저조도(low lillumination) 및 노이즈(noise)에 대해 명암도와 노이즈 개선이 필요하며 이를 위해 Gaussian Blur와 Sharpening filtering을 통해 노이즈 감소 및 이미지 선명도를 높이는 작업 수행
    - 안개 환경의 대비(contrast) 감소와 색상 변화로 인해 대비 향상 및 color balance를 조정하여 이미지 선명도를 높이는 작업 수행(다양한 Filtering 적용)
    - 강우 환경의 빗물로 인한 Camera blur 처리는 second-order derivatives의 Laplacian filter를 적용하여 경계선(edge) 추출을 통해 경계선의 선명도를 분석하여 강우 상황을 판단할 수 있도록 진행
    - 역광 환경의 대비(contrast) 감소와 음영(shadow) 발생으로 인해 대비 향상 및 음영 필터링을 위한 히스토그램 평활화(Histogram Equalization) 및 CLAHE Filtering을 통해 이미지 선명도를 높이는 작업 수행
    - 급격한 조도 변화 환경은 히스토그램 평활화(Histogram Equalization) 및 CLAHE Filtering을 통해 이미지 선명도를 높이는 작업 수행

Midterm Project

- 프로젝트 진행
  - 야간 환경 이미지 필터링
    - Gaussian Blur 및 Sharpening Filtering을 통해 노이즈 감소 및 이미지 선명도 개선
    - bad_weather_night.py

bad_weather_image_filtering.py | bad_weather_night.py

```python
import cv2
import numpy as np

# image set-up(1920*1080)
image = cv2.imread('/home/jinhakim/anaconda3/envs/opencv_lec/Midterm_Project/bad_weather_images/night/16_211922_220929_40.jpg')

# Applying Gaussian Blur to image
gaussian_blur = cv2.GaussianBlur(image, ksize: (7,7), sigmaX: 0)

# Applying Sharpening filter
sharpening = cv2.addWeighted(image, alpha: 2.8, gaussian_blur, -0.6, gamma: 0)

# Visualization of applying filtering
cv2.imshow( winname: 'Filtering Applied Image', sharpening)
cv2.waitKey()
cv2.destroyAllWindows()
```

- 프로젝트 진행
  - 야간 환경 이미지 필터링(명암 및 노이즈 개선)

원본 이미지

필터링 적용 이미지

Midterm Project

- 프로젝트 진행
  - 강우 환경 이미지 필터링
    - Laplacian Filtering을 통해 빗물로 인한 Blur 제거
    - bad_weather_rain.py

bad_weather_image_filtering.py | bad_weather_night.py | bad_weather_fog.py | bad_weather_rain.py

```python
import cv2
import numpy as np

# image set-up
image = cv2.imread('/home/jinhakim/anaconda3/envs/opencv_lec/Midterm_Project/bad_weather_images/rain/24_111049_220829_01.jpg')

# Converting image color(BGR) to grayscale
gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

# Applying Laplacian filter to bad weather image(blurred)
laplacian_filtering = cv2.Laplacian(gray_image, cv2.CV_64F)

# Applying absolute value and converting type
laplacian_filtering = np.uint8(np.absolute(laplacian_filtering))

# Visualization of applying Laplacian filter
cv2.imshow( winname: 'Laplacian Filter Applied Image', laplacian_filtering)
cv2.waitKey()
cv2.destroyAllWindows()
```

Midterm Project

- 프로젝트 진행
  - 강우 환경 이미지 필터링(빗방울로 인한 Blur 제거)

원본 이미지

Laplacian Filtering

Midterm Project

- 프로젝트 진행
  - 안개 환경 이미지 필터링-01
    - Gaussian Blur, Median Blur, Bilateral Filter를 통해 이미지 선명도 개선
    - bad_weather_fog.py

bad_weather_image_filtering.py | bad_weather_night.py | bad_weather_fog.py

```python
import cv2
import matplotlib.pyplot as plt

# image set-up(1920*1080)
image = cv2.imread('/home/jinhakim/anaconda3/envs/opencv_lec/Midterm_Project/bad_weather_images/fog/16_072312_221021_08.jpg')

# Gaussian Blur
gaussian_blur = cv2.GaussianBlur(image, ksize: (5, 5), sigmaX: 0)

# Median Blur
median_blur = cv2.medianBlur(image, ksize: 5)

# Bilateral Filter
bilateral_filter = cv2.bilateralFilter(image, d: 9, sigmaColor: 75, sigmaSpace: 75)

# Visualization of Applying filtering
titles = ['original image', 'gaussian image', 'median image', 'bilateral image']
images = [image, gaussian_blur, median_blur, bilateral_filter]

for i in range(4):
    plt.subplot(*args: 2, 2, i+1)
    plt.imshow(images[i], cmap='gray')
    plt.title(titles[i])
    plt.xticks([]), plt.yticks([])

plt.tight_layout()
plt.show()
```

❖ 프로젝트 진행

➢ 안개 환경 이미지 필터링-01(각 필터링 별 육안상 뚜렷한 차이는 없음.)

원본 이미지

Gaussian Blur

Median Blur

Bilateral Filter

Midterm Project

- 프로젝트 진행
  - 안개 환경 이미지 필터링-02
    - Laplacian Filtering을 통해 안개(Blur) 제거
    - bad_weather_fog_v2.py

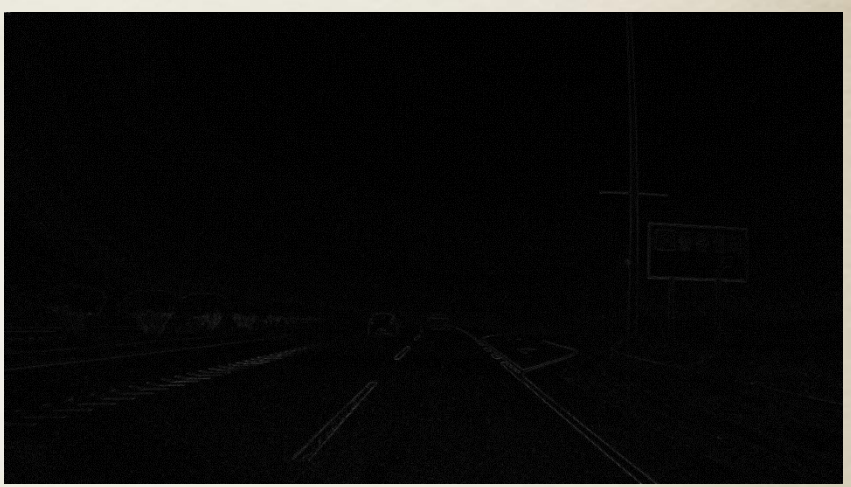
bad_weather_fog_v2.py | bad_weather_rain.py | bad_weather_backlight_v2.py | bad_weather_tunnel_v2.py | bad_weather_tunnel_v

```python
import cv2
import numpy as np

# image set-up
image = cv2.imread('/home/jinhakim/anaconda3/envs/opencv_lec/Midterm_Project/bad_weather_images/fog/16_072312_221021_08.jpg')

# Converting image color(BGR) to grayscale
gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

# Applying Laplacian filter to bad weather image(blurred)
laplacian_filtering = cv2.Laplacian(gray_image, cv2.CV_64F)

# Applying absolute value and converting type
laplacian_filtering = np.uint8(np.absolute(laplacian_filtering))

# Visualization of applying Laplacian filter
cv2.imshow( winname: 'Laplacian Filter Applied Image', laplacian_filtering)
cv2.waitKey()
cv2.destroyAllWindows()
```

❖ 프로젝트 진행

➢ 안개 환경 이미지 필터링-02(Laplacian Filtering 적용 시)

원본 이미지



Laplacian Filtering

▪ 안개를 Blur로 간주하면 Laplacian Filtering 적용이 Blur 제거에 효과가 있는 것으로 판단됨.

Midterm Project

- 프로젝트 진행
  - 안개 환경 이미지 필터링-03
    - Canny Edge Filtering을 통해 안개(Blur) 제거
    - bad_weather_fog_v3.py

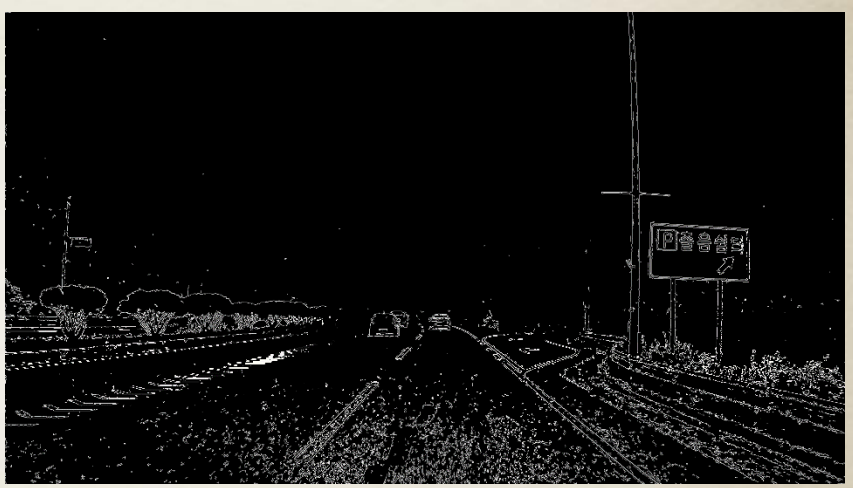
bad_weather_rain.py | bad_weather_backlight_v2.py | bad_weather_tunnel_v2.py | bad_weather_tunnel_v1.py | bad_weather_fog_v3.py

```python
import cv2

# image set-up(1920*1080)
image = cv2.imread('/home/jinhakim/anaconda3/envs/opencv_lec/Midterm_Project/bad_weather_images/fog/16_072312_221021_08.jpg')

# Converting image color(BGR) to grayscale
gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

# Applying Canny Edge Filtering
edges = cv2.Canny(gray_image, threshold1=30, threshold2=60)

# Visualization of applying filtering
cv2.imshow( winname: 'Canny Edge Filtering', edges)
cv2.waitKey()
cv2.destroyAllWindows()
```

❖ 프로젝트 진행

➢ 안개 환경 이미지 필터링-03(Canny Edge Filtering 적용 시)

원본 이미지



Canny Edge Filtering

- 안개를 Blur로 간주하면 Canny Edge Filtering 적용이 Blur 제거에 효과가 있는 것으로 판단됨.
- 다만, 결과 이미지의 노이즈를 제거하기 위한 방법도 고민할 필요가 있음.

Midterm Project

- 프로젝트 진행
  - 역광 환경 이미지 필터링-01
    - 히스토그램 평활화(Histogram Equalization)를 통해 이미지 전체 밝기를 조절하여 명암 차이를 감소시켜 주변 배경 및 객체 선명도 향상
    - bad_weather_backlight_v1.py

bad_weather_image_filtering.py | bad_weather_backlight_v1.py | bad_weather_night.py | bad_weather_fog.py | bad_weather_rain.py

```python
import cv2

# image set-up(1920*1080)
image = cv2.imread('/home/jinhakim/anaconda3/envs/opencv_lec/Midterm_Project/bad_weather_images/reverse_light/16_080557_221019_01.jpg')

# Converting image color(BGR) to grayscale
gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

# Applying histogram equalization
equalized_image = cv2.equalizeHist(gray_image)

# Converting equalized_image to BGR type
equalized_image_bgr = cv2.cvtColor(equalized_image, cv2.COLOR_GRAY2BGR)

# Visualization of applying filtering
cv2.imshow( winname: 'Backlight Corrected Image', equalized_image_bgr)

cv2.waitKey(0)
cv2.destroyAllWindows()
```

- 프로젝트 진행
  - 역광 환경 이미지 필터링-01(대비 향상 및 음영 감소)

원본 이미지

Histogram Equalization

Midterm Project

- 프로젝트 진행
  - 역광 환경 이미지 필터링-02
    - CLAHE(Contrast Limited Adaptive Histogram Equalization)를 통해 이미지의 contrast 및 주변 객체의 선명도 향상(수업에서 다루지 않은 기법으로 관련 code는 google에서 찾아 적용)
    - bad_weather_backlight_v2.py

bad_weather_night.py | bad_weather_fog.py | bad_weather_rain.py | bad_weather_backlight_v1.py | bad_weather_backlight_v2.py

```python
import cv2

# image set-up(1920*1080)
image = cv2.imread('/home/jinhakim/anaconda3/envs/opencv_lec/Midterm_Project/bad_weather_images/reverse_light/16_082602_221022_35.jpg')

# Reducing shadow
clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8, 8))
clahe_image = cv2.cvtColor(image, cv2.COLOR_BGR2LAB)
clahe_image[:, :, 0] = clahe.apply(clahe_image[:, :, 0])
shadow_reduced = cv2.cvtColor(clahe_image, cv2.COLOR_LAB2BGR)

# Enhancing contrast
lab = cv2.cvtColor(shadow_reduced, cv2.COLOR_BGR2LAB)
l, a, b = cv2.split(lab)
clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8, 8))
l = clahe.apply(l)
lab = cv2.merge((l, a, b))
contrast_enhanced = cv2.cvtColor(lab, cv2.COLOR_LAB2BGR)

# Visualization of applying flitering
cv2.imshow( winname: 'CLAHE Filter Applied Image', contrast_enhanced)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

- 프로젝트 진행
  - 역광 환경 이미지 필터링-02(대비 향상 및 음영 감소)

원본 이미지

CLAHE Filtering

Midterm Project

- 프로젝트 진행
  - 급격한 조도 변화 환경 이미지 필터링-01
    - 히스토그램 평활화(Histogram Equalization)를 통해 이미지 전체 밝기를 조절하여 명암 차이를 감소시켜 주변 배경 및 객체 선명도 향상
    - bad_weather_tunnel_v1.py

bad_weather_fog.py | bad_weather_rain.py | bad_weather_backlight_v2.py | bad_weather_tunnel_v2.py | bad_weather_tunnel_v1.py

```python
import cv2

# image set-up(1920*1080)
image = cv2.imread('/home/jinhakim/anaconda3/envs/opencv_lec/Midterm_Project/bad_weather_images/tunnel/24_111849_220829_38.jpg')

# Converting image color(BGR) to grayscale
gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

# Applying histogram equalization
equalized_image = cv2.equalizeHist(gray_image)

# Converting equalized_image to BGR type
equalized_image_bgr = cv2.cvtColor(equalized_image, cv2.COLOR_GRAY2BGR)

# Visualization of applying filtering
cv2.imshow( winname: 'Corrected Image', equalized_image_bgr)

cv2.waitKey(0)
cv2.destroyAllWindows()
```

Midterm Project

❖ 프로젝트 진행

➢ 급격한 조도 변화 환경 이미지 필터링-01

원본 이미지

Histogram Equalization

- Histogram Equalization을 적용했을 때 오히려 원본 이미지보다 터널 안/밖 밝기 조절이 떨어지는 결과가 나왔음.

Midterm Project

- 프로젝트 진행
  - 급격한 조도 변화 환경 이미지 필터링-02
    - CLAHE(Contrast Limited Adaptive Histogram Equalization)를 통해 이미지의 contrast 및 주변 객체의 선명도 향상(수업에서 다루지 않은 기법으로 관련 code는 google에서 찾아 적용)
    - bad_weather_tunnel_v2.py

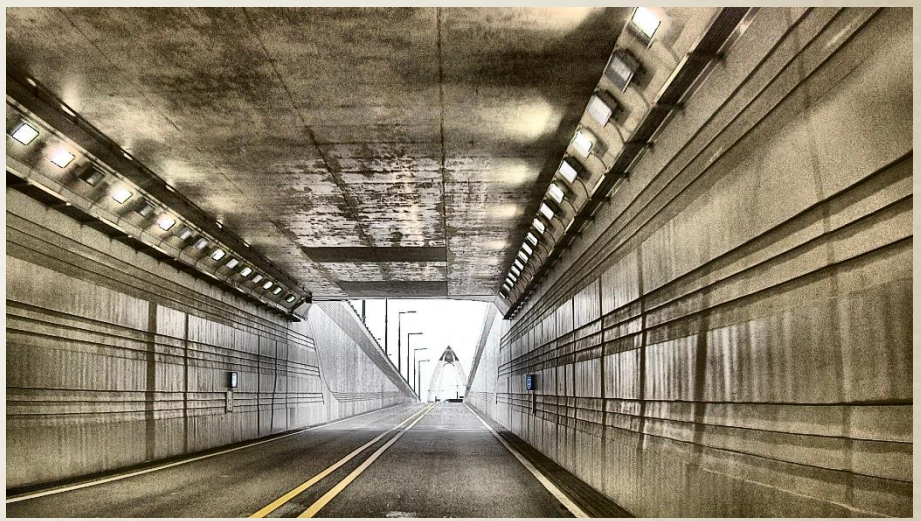
bad_weather_fog.py | bad_weather_rain.py | bad_weather_backlight_v2.py | bad_weather_tunnel_v2.py | bad_weather_tunnel_v1.py

```python
import cv2

# image set-up(1920*1080)
image = cv2.imread('/home/jinhakim/anaconda3/envs/opencv_lec/Midterm_Project/bad_weather_images/tunnel/24_111849_220829_38.jpg')

# Reducing shadow
clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8, 8))
clahe_image = cv2.cvtColor(image, cv2.COLOR_BGR2LAB)
clahe_image[:, :, 0] = clahe.apply(clahe_image[:, :, 0])
shadow_reduced = cv2.cvtColor(clahe_image, cv2.COLOR_LAB2BGR)

# Enhancing contrast
lab = cv2.cvtColor(shadow_reduced, cv2.COLOR_BGR2LAB)
l, a, b = cv2.split(lab)
clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8, 8))
l = clahe.apply(l)
lab = cv2.merge((l, a, b))
contrast_enhanced = cv2.cvtColor(lab, cv2.COLOR_LAB2BGR)

# Visualization of applying flitering
cv2.imshow( winname: 'CLAHE Filter Applied Image', contrast_enhanced)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

❖ 프로젝트 진행

➢ 급격한 조도 변화 환경 이미지 필터링-01

원본 이미지



CLAHE Filtering

▪ CLAHE Filtering을 적용했을 때 원본 이미지보다 터널 안/밖 밝기 조절이 향상되는 결과가 나왔음.

Midterm Project

- 프로젝트 결과 고찰
  - 야간, 강우, 역광, 급격한 조도 변화 환경에서의 이미지는 대체로 적합한 filtering을 통해 이미지 개선이 되는 것을 확인할 수 있었음.
  - 안개의 경우 여러 가지 filtering 기법을 적용하였고 Gaussian, Median, Bilateral Filtering은 큰 차이는 없었고, 안개를 Blur라고 가정했을 때 Blur를 제거한 Laplacian이나 Canny Edge Filtering이 적절할 것으로 판단되나 더 적합한 Filtering 기법에 대한 추후 연구가 필요함.

THANK YOU

AVGenius

AVGenius