산업 컴퓨터비젼 실제 Mid-term Project

산업인공지능학과
2020254018 강윤구

1. 주제

1) 현업에서 컴퓨터비전 관련 문제 : 전산실의 서버에 대한 Power on/off 인식

2) 수집 사진 : 초록색 LED는 Power on, 주황색 LED는 Power off 로 구분

3) Equalize Histogram 확인

4) Canny Edge 확인

5) Mouse로 ROI 지정하여 Threshold 확인

파일명 : Server-6.png

2. 실행결과 #1 (Equalize Histogram 확인)

```python
import cv2
#import argparse
#import random
import numpy as np
import matplotlib.pyplot as plt


grey = cv2.imread('../Assignment/data/Server-6.png', 0)
image = cv2.imread('../Assignment/data/Server-6.png')


# Threshold 값 입력
in_thr = np.int(input("Input Threshold (ex:200) : "))
image_eq = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
thr, mask = cv2.threshold(image_eq, in_thr, 1, cv2.THRESH_BINARY)
print('Threshole used:', thr)


# Equalize Histogram
cv2.imshow('original grey', grey)
#cv2.waitKey()

hist, bins = np.histogram(grey, 256, [0, 255])
plt.fill(hist)
plt.xlabel('pixel value')
plt.show()

grey_eq = cv2.equalizeHist(grey)
hist, bins = np.histogram(grey_eq, 256, [0, 255])
plt.fill_between(range(256), hist, 0)
plt.xlabel('pixel value')
plt.show()

cv2.imshow('equalized grey', grey_eq)
cv2.waitKey()

hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)

hsv[..., 2] = cv2.equalizeHist(hsv[..., 2])
color_eq = cv2.cvtColor(hsv, cv2.COLOR_HSV2BGR)
cv2.imshow('original color', image)
cv2.imshow('equalized color', color_eq)
cv2.waitKey()
cv2.destroyAllWindows()
```

Grey

Color

2. 실행결과 #2 (Canny Edge 확인)

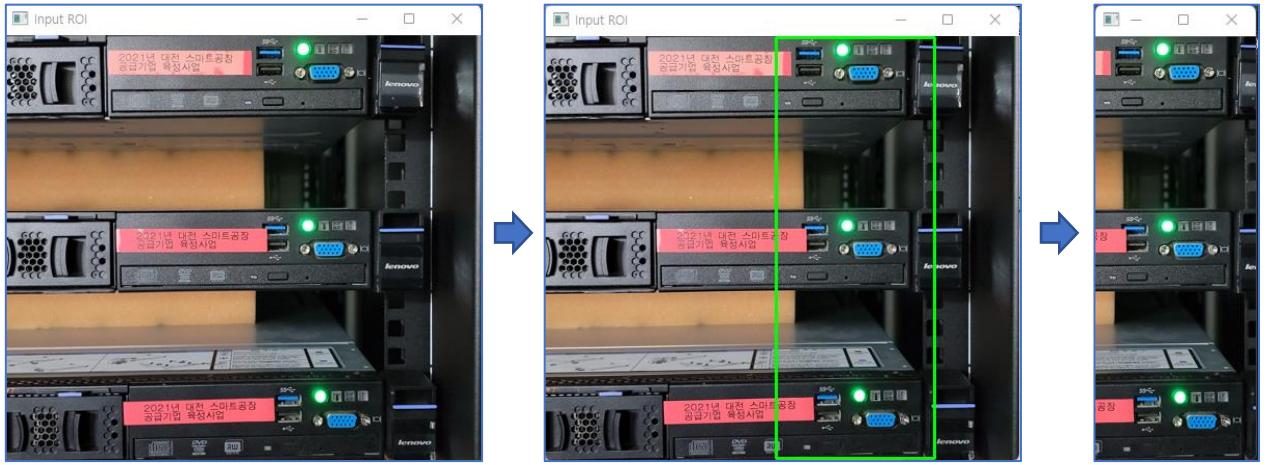
```python
# Canny Edge
def canny():
#     edge1 = cv2.Canny(grey, 50, 200)
#     edge2 = cv2.Canny(grey, 100, 200)
    edge3 = cv2.Canny(grey, 170, 200)

    cv2.imshow('original', grey)
#     cv2.imshow('Canny Edge1', edge1)
#     cv2.imshow('Canny Edge2', edge2)
    cv2.imshow('Canny Edge3', edge3)

    cv2.waitKey(0)
    cv2.destroyAllWindows()

canny()
```

Canny Edge (minVal : 170, maxVal 200)

2. 실행결과 #3 (Mouse로 ROI 지정하여 Threshold 확인)

```python
# Mouse로 ROI 지정
image_to_show = np.copy(image)
w, h = image.shape[1], image.shape[0]

mouse_pressed = False
s_x = s_y = e_x = e_y = -1

def mouse_callback(event, x, y, flags, param):
    global image_to_show, s_x, s_y, e_x, e_y, mouse_pressed

    if event == cv2.EVENT_LBUTTONDOWN:
        mouse_pressed = True
        s_x, s_y = x, y
        image_to_show = np.copy(image)

    elif event == cv2.EVENT_MOUSEMOVE:
        if mouse_pressed:
            image_to_show = np.copy(image)
            cv2.rectangle(image_to_show, (s_x, s_y), (x, y), (0, 255, 0), 2)

    elif event == cv2.EVENT_LBUTTONUP:
        mouse_pressed = False
        e_x, e_y = x, y

cv2.namedWindow('Input ROI')
cv2.setMouseCallback('Input ROI', mouse_callback)

while True:
    cv2.imshow('Input ROI', image_to_show)
    k = cv2.waitKey(1)

    if k == ord('c'):
        if s_y > e_y:
            s_y, e_y = e_y, s_y
        if s_x > e_x:
            s_x, e_x = e_x, s_x

        if e_y - s_y > 1 and e_x - s_x > 0:
            image = image[s_y:e_y, s_x:e_x]
            image_to_show = np.copy(image)
    elif k == 27:
        break

cv2.destroyAllWindows()


# ROI 부분에 대한 Threshold
image_eq = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

thr, mask = cv2.threshold(image_eq, in_thr, 1, cv2.THRESH_BINARY)

plt.figure(figsize=(10, 4))
plt.subplot(121)
plt.axis('off')
plt.title('ROI image')
plt.imshow(image_eq, cmap='gray')
plt.subplot(122)
plt.axis('off')
plt.title('binary threshold')
plt.imshow(mask, cmap='gray')
plt.tight_layout()
plt.show()

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

Mouse로 ROI 지정



Threshold 값 250으로 하여 확인(LED 부분 흰색으로 확인)

3. 결론

1) 이미지에 따라 Threshold 값을 다르게 적용해서 최상의 화면을 찾을 수 있다.

2) Mouse를 이용하여 ROI 지정 후 Threshold 하여 원하는 결과값을 확인 할 수 있다.

3) 결과값을 기계학습 하면 서버의 Power on/off 를 구분 할 수 있을 것 같다.