인공지능 응용시스템

License Plate Recognition

12141755 임현호
12141725 안수진

목차

목차

01

License Plate Recognition

01. 개요

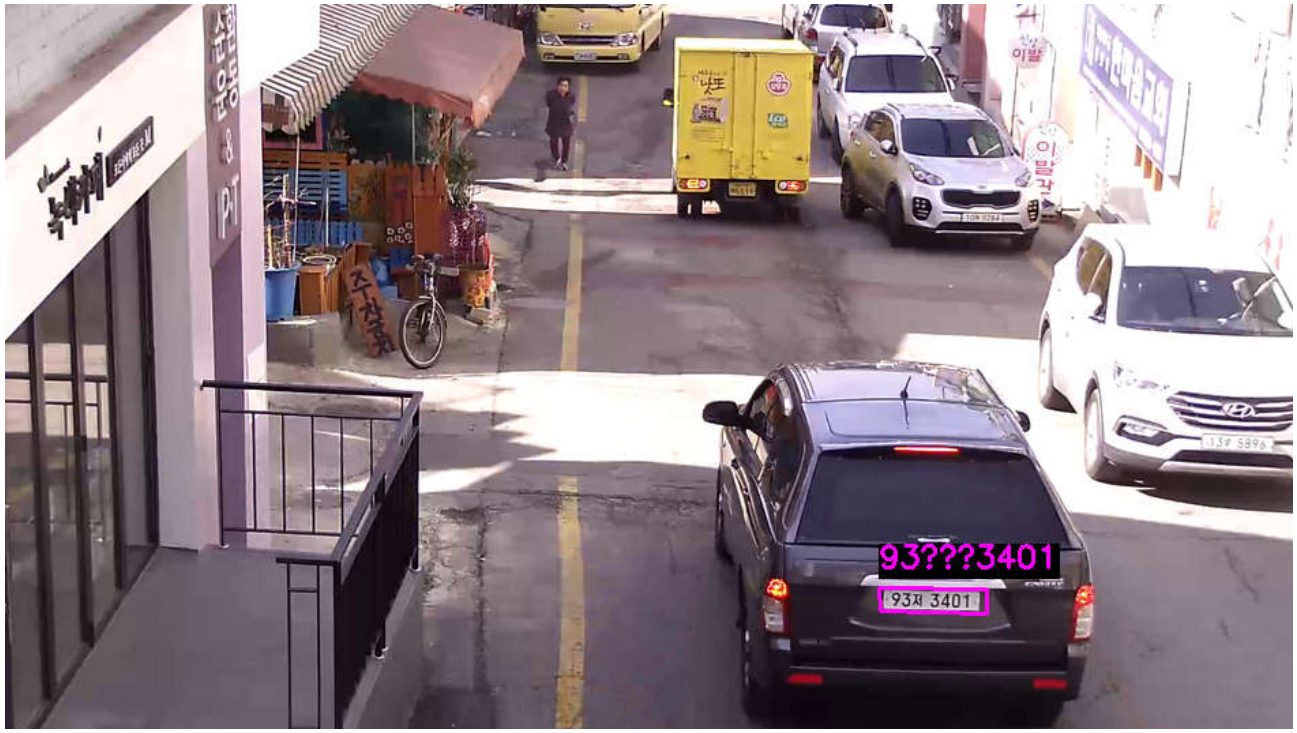
License Plate Recognition

번호판 인식

01. 개요

License Plate Recognition

번호판 인식



01. 개요

License Plate Recognition

번호판 종류

Type	License Plate	Type
1		P1
2		P2
3		P3
4		P4
5		P5
6		P6

01. 개요

License Plate Recognition

점수 계산

$$Score = Score_{park} + Score_{cctv} + 0.1 \times (100 - PT)$$

$$PT = m\,\text{sec.}/image\,(average)$$

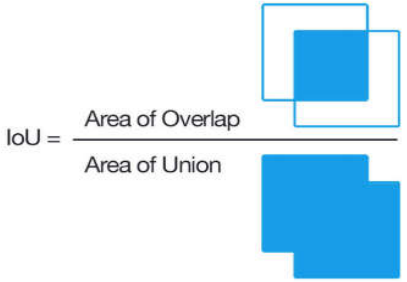
$$Score_i = Accuracy_{det} + Accuracy_{rec} \quad (i = park\ or\ cctv)$$

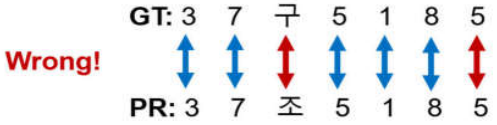
$$Accuracy_{det} = \frac{1}{n}\sum_{i=0}^{n-1}\frac{\#TP_{det} - \#FP_{det}}{\#GT} \times 100\%$$

$$Accuracy_{rec} = \frac{1}{n}\sum_{i=0}^{n-1}\frac{\#TP_{rec}}{\#GT} \times 100\%$$

- PT: average processing time of the model (unit: msec.)
- #TP_{det}: number of true positive for detection
- #FP_{det}: number o false positive for detection
- #TP_{rec}: number of true positive for recognition
- #GT: number of ground-truth

- TP_{det}: IoU >= θ, θ=0.7
- FP_{det}: IoU < θ





목차

02

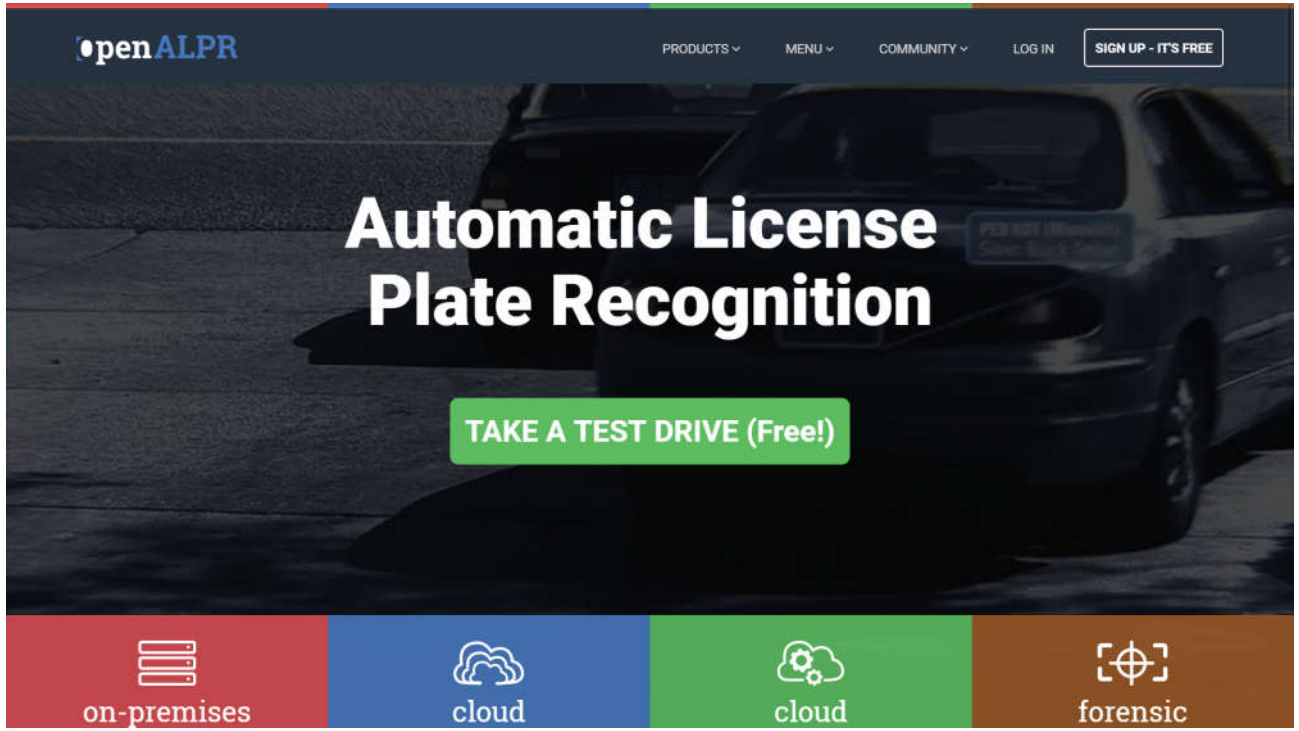
OpenAlpr

02. 방법

OpenAlpr

홈페이지

http://www.openalpr.com/

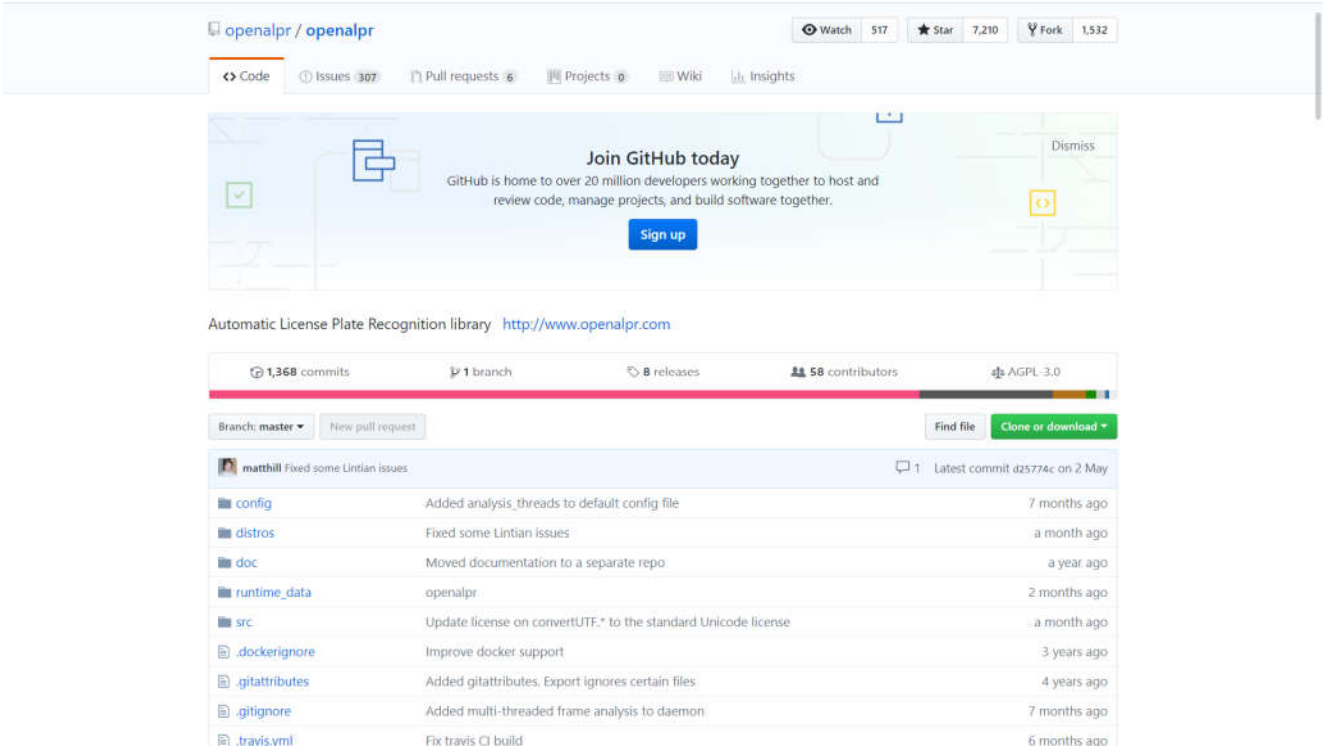


02. 방법

OpenAlpr

GitHub

https://github.com/openalpr/openalpr



02. 방법

OpenAlpr

실행 예시


```
hyunho@hyunho-13ZD940-GX50K:~/Desktop$ alpr -c kr 000189.jpg
plate0: 2 results
    - 16수0944    confidence: 94.9764
    - 16주0944    confidence: 81.4146
```

```
hyunho@hyunho-13ZD940-GX50K:~/Desktop/aitest/parking/img_gt_1$ alpr -c kr --json
 000189.jpg
{"version":2,"data_type":"alpr_results","epoch_time":1528960839424,"img_width":1
600,"img_height":1200,"processing_time_ms":340.26947021484375,"uuid":"","error":
false,"regions_of_interest":[{"x":0,"y":0,"width":1600,"height":1200}],"results"
:[{"plate":"16수0944","confidence":94.9721908569336,"matches_template":1,"plate_
index":0,"region":"kr","region_confidence":0,"processing_time_ms":46.89348602294
9219,"requested_topn":10,"coordinates":[{"x":985,"y":257},{"x":1322,"y":271},{"x
":1317,"y":352},{"x":994,"y":336}],"vehicle_region":{"x":864,"y":0,"width":575,"
height":575},"candidates":[{"plate":"16수0944","confidence":94.9721908569336,"ma
tches_template":1},{"plate":"16주0944","confidence":81.411079406738281,"matches_
template":1}]}]}
```

02. 방법

OpenAlpr

실행 예시

```
import ctypes
import json
import platform

# We need to do things slightly differently for Python 2 vs. 3
# ... because the way str/unicode have changed to bytes/str
if platform.python_version_tuple()[0] == '2':
     # Using Python 2
     bytes = str
     _PYTHON_3 = False
else:
     # Assume using Python 3+
     unicode = str
     _PYTHON_3 = True

def _convert_to_charp(string):
     # Prepares function input for use in c-functions as char*
     if type(string) == unicode:
          return string.encode("UTF-8")
     elif type(string) == bytes:
          return string
     else:
          raise TypeError("Expected unicode string values or ascii/bytes values. Got: %r" % type
(string))

def _convert_from_charp(charp):
     # Prepares char* output from c-functions into Python strings
     if _PYTHON_3 and type(charp) == bytes:
          return charp.decode("UTF-8")
     else:
          return charp

class Alpr():
     def __init__(self, country, config_file, runtime_dir):
          """
          Initializes an OpenALPR instance in memory.

          :param country: The default region for license plates. E.g., "us" or "eu"
          :param config_file: The path to the OpenALPR config file
          :param runtime_dir: The path to the OpenALPR runtime data directory
          :return: An OpenALPR instance
          """
          country = _convert_to_charp(country)
          config_file = _convert_to_charp(config_file)
          runtime_dir = _convert_to_charp(runtime_dir)
          try:
          # Load the .dll for Windows and the .so for Unix-based
               if platform.system().lower().find("windows") != -1:
                    self._openalprpy_lib = ctypes.cdll.LoadLibrary("libopenalprpy.dll")
               elif platform.system().lower().find("darwin") != -1:
                    self._openalprpy_lib = ctypes.cdll.LoadLibrary("libopenalprpy.dylib")
               else:
                    self._openalprpy_lib = ctypes.cdll.LoadLibrary("libopenalprpy.so")
          except OSError as e:
               nex = OSError("Unable to locate the OpenALPR library. Please make sure that OpenALPR is
```

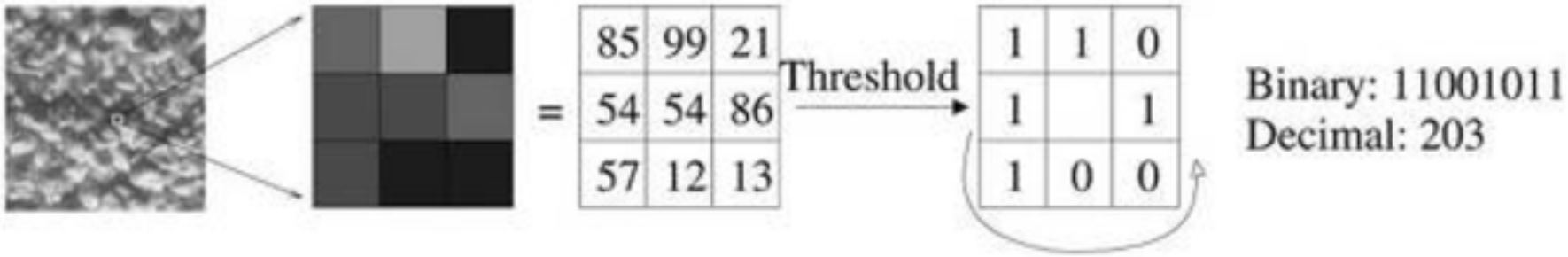
02

Plate Detection

02. 방법

Plate Detection

Local Binary Pattern Algorithm

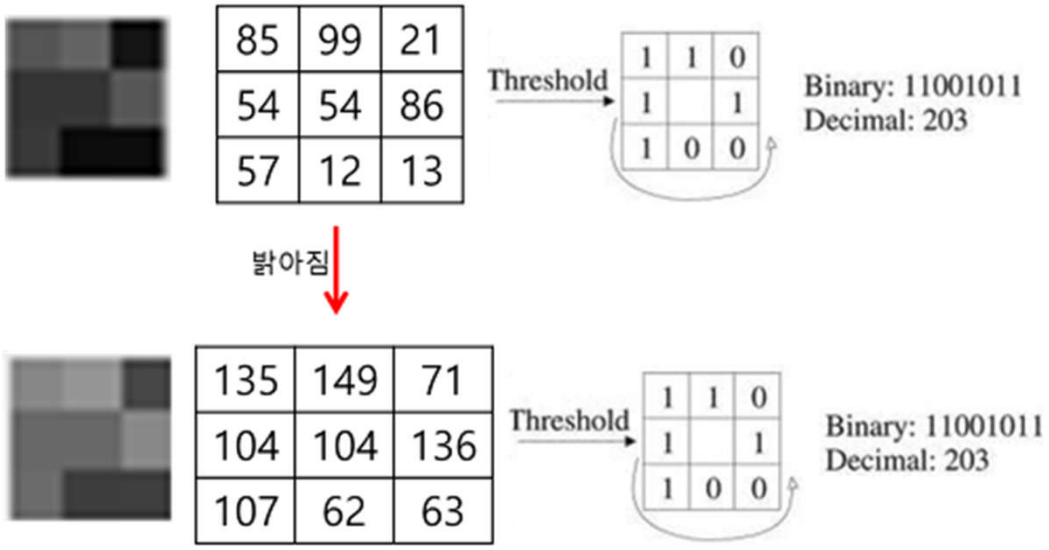


T. Ahonen, A. Hadid, and M. Pietikinen, "Face description with local binary patterns: Application to face recognition," PAMI 2006.

02. 방법

Plate Detection

Local Binary Pattern Algorithm



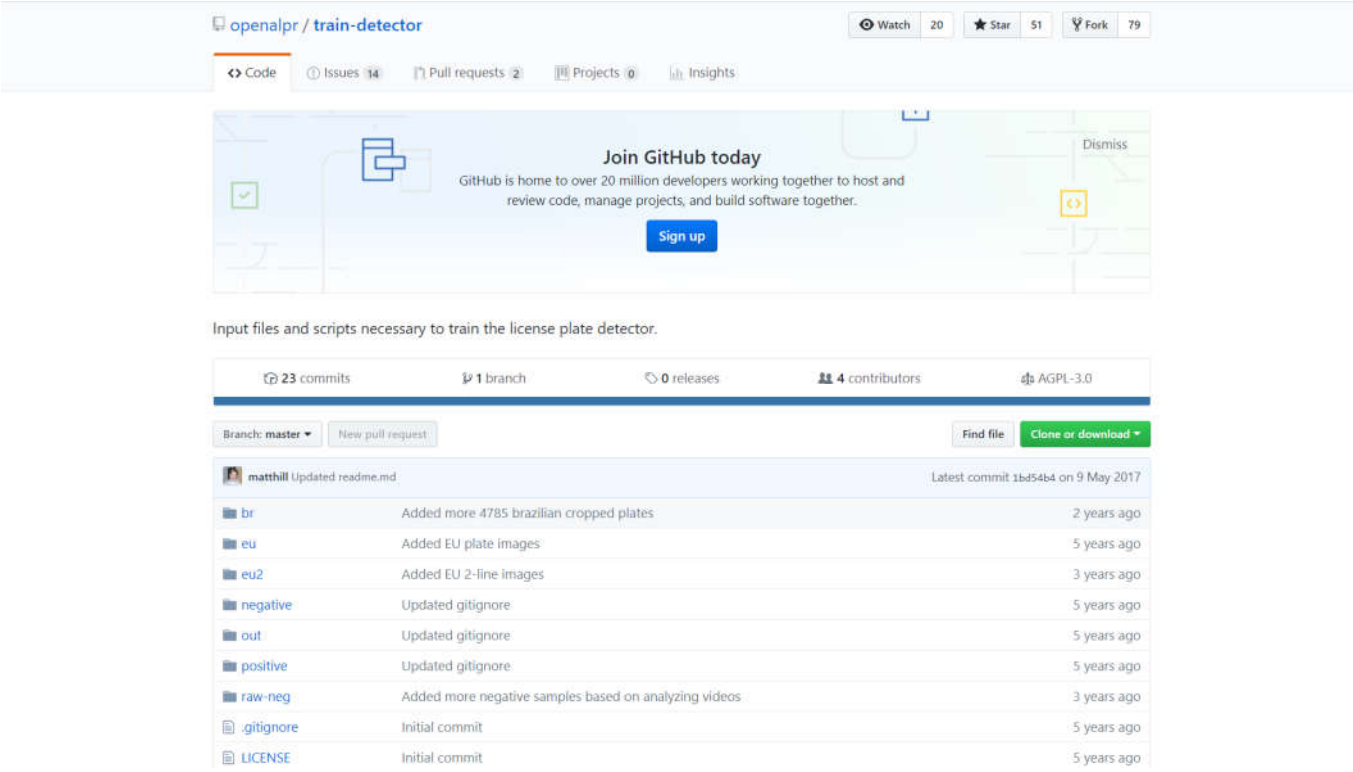
T. Ahonen, A. Hadid, and M. Pietikinen, "Face description with local binary patterns: Application to face recognition," PAMI 2006.

02. 방법

Plate Detection

train-detector

https://github.com/openalpr/train-detector



02

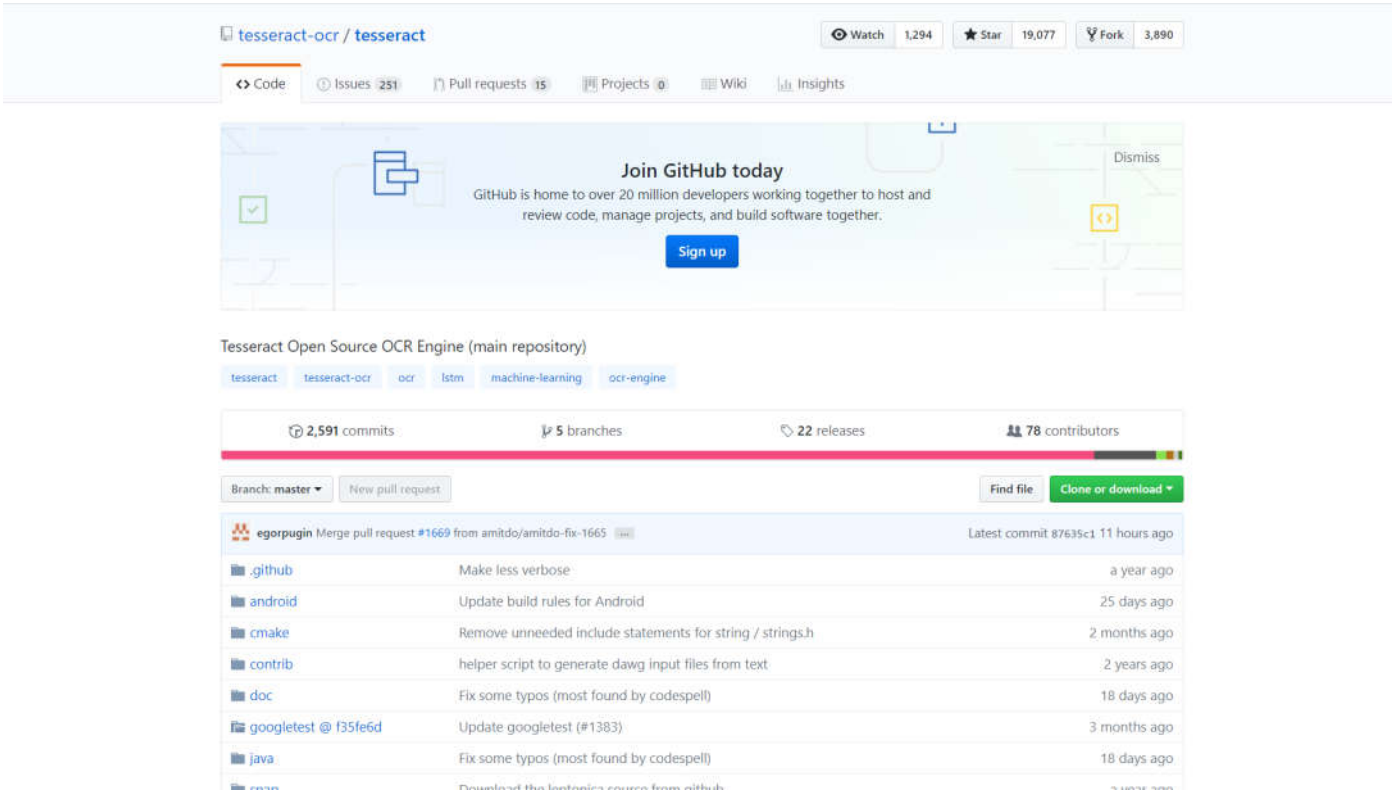
Plate Recognition

02. 방법

Plate Recognition

Tesseract

https://github.com/tesseract-ocr/tesseract



02. 방법

Plate Recognition

Tesseract

Fig. 1. An example of a curved fitted baseline.

R. Smith, “An overview of the Tesseract OCR engine. ,” in Int. Conf. on Document Analysis and Recognition (ICDAR), Curitiba, Brazil, 2007.

02. 방법

Plate Recognition

Tesseract

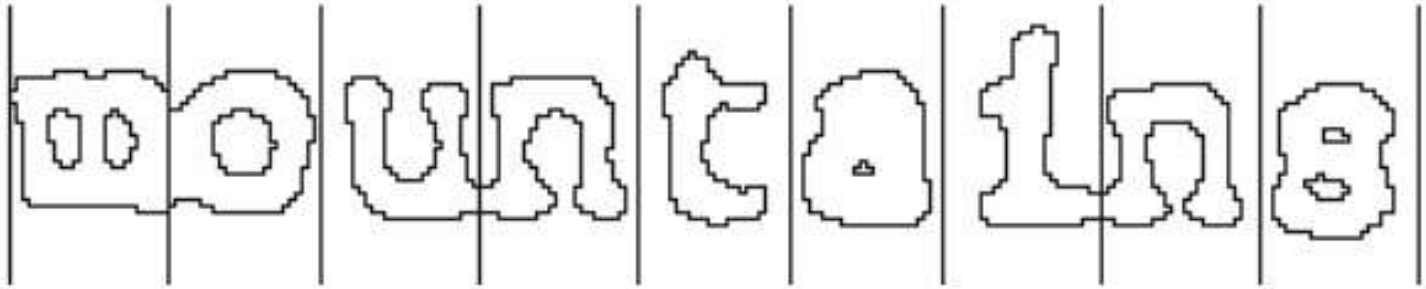


Fig. 2. A fixed-pitch chopped word.

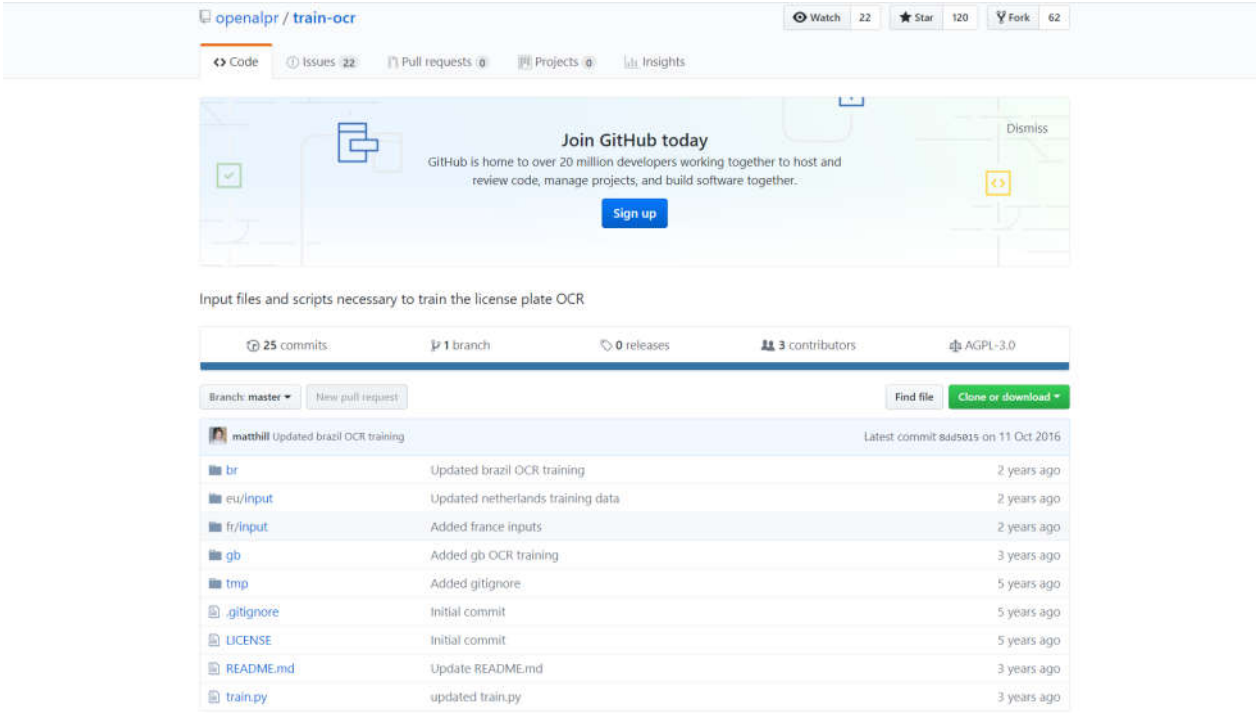
R. Smith, “An overview of the Tesseract OCR engine. ,” in Int. Conf. on Document Analysis and Recognition (ICDAR), Curitiba, Brazil, 2007.

02. 방법

Plate Recognition

train-ocr

https://github.com/openalpr/train-ocr



02. 방법

Plate Recognition

번호판 규격

☐ 제5조(차종 및 용도구분등의 기호) ① 등록번호판의 차종 및 용도별 분류기호를 다음과 같이 한다.

구 분		분 류	기 호
차종별		승용자동차	01-69
		승합자동차	70-79
		화물자동차	80-97
		특수자동차	98,99
용도별	비사업용 (SOFA 자동차 포함)	자가용 (관용포함)	가,나,다,라,마,거,너,더,러,머,버,서,어,저,고,노,도,로,모,보,소,오,조,구,누,두,루,무,부,수,우,주
	자동차운수 사업용	일반용	바, 사, 아, 자, 배
		대여사업용	허, 하, 호
	외교용	외교관용	외교
		영사용	영사
		준외교관용	준외
		준영사용	준영
		국제기구용	국기
		기타외교용	협정, 대표

② 이륜자동차번호판은 가·나·다·라·라·마·바·사·아·자·차·카·타·파·하를 용도별 기호로서 표시한다.

☐ 제6조(관할관청 기호표시) 관할관청의 기호표시는 다음과 같이 한다. 다만, 비사업용 판에는 관할 시·군 또는 구의 명칭을 함께 표시한다.

서울특별시 : 서울
부산광역시 : 부산
대구광역시 : 대구
인천광역시 : 인천
광주광역시 : 광주
대전광역시 : 대전
울산광역시 : 울산
세종특별자치시 : 세종
경기도 : 경기
강원도 : 강원
충청북도 : 충북
충청남도 : 충남
전라북도 : 전북
전라남도 : 전남
경상북도 : 경북
경상남도 : 경남
제주도 : 제주

http://law.go.kr/%ED%96%89%EC%A0%95%EA%B7%9C%EC%B9%99/%EC%9E%90%EB%8F%99%EC%B0%A8%20%EB%93%B1%EB%A1%9D%EB%B2%88%ED%98%B8%ED%8C%90%20%EB%93%B1%EC%9D%98%20%EA%B8%B0%EC%A4%80%EC%97%90%20%EA%B4%80%ED%95%9C%20%EA%B3%A0%EC%8B%9C

목차

03

Preprocess

03. 구현

Preprocess

번호판 좌표 수정

03. 구현

Preprocess

번호판 위치 조정

03. 구현

Preprocess

Detection 크기 조정

```
; detection will ignore plates that are too large.  This is a good efficiency technique to use if the
; plates are going to be a fixed distance away from the camera (e.g., you will never see plates that
fill
; up the entire image
max_plate_width_percent = 100
max_plate_height_percent = 100

; detection_iteration_increase is the percentage that the LBP frame increases each iteration.
; It must be greater than 1.0.  A value of 1.01 means increase by 1%, 1.10 increases it by 10% each
time.
; So a 1% increase would be ~10x slower than 10% to process, but it has a higher chance of landing
; directly on the plate and getting a strong detection
detection_iteration_increase = 1.10

; The minimum detection strength determines how sure the detection algorithm must be before signaling
that
; a plate region exists.  Technically this corresponds to LBP nearest neighbors (e.g., how many
detections
; are clustered around the same area).  For example, 2 = very lenient, 9 = very strict.
detection_strictness = 2

; The detection doesn't necessarily need an extremely high resolution image in order to detect plates
; Using a smaller input image should still find the plates and will do it faster
; Tweaking the max_detection_input values will resize the input image if it is larger than these
sizes
; max_detection_input_width/height are specified in pixels
max_detection_input_width = 1280
max_detection_input_height = 720
```

03. 구현

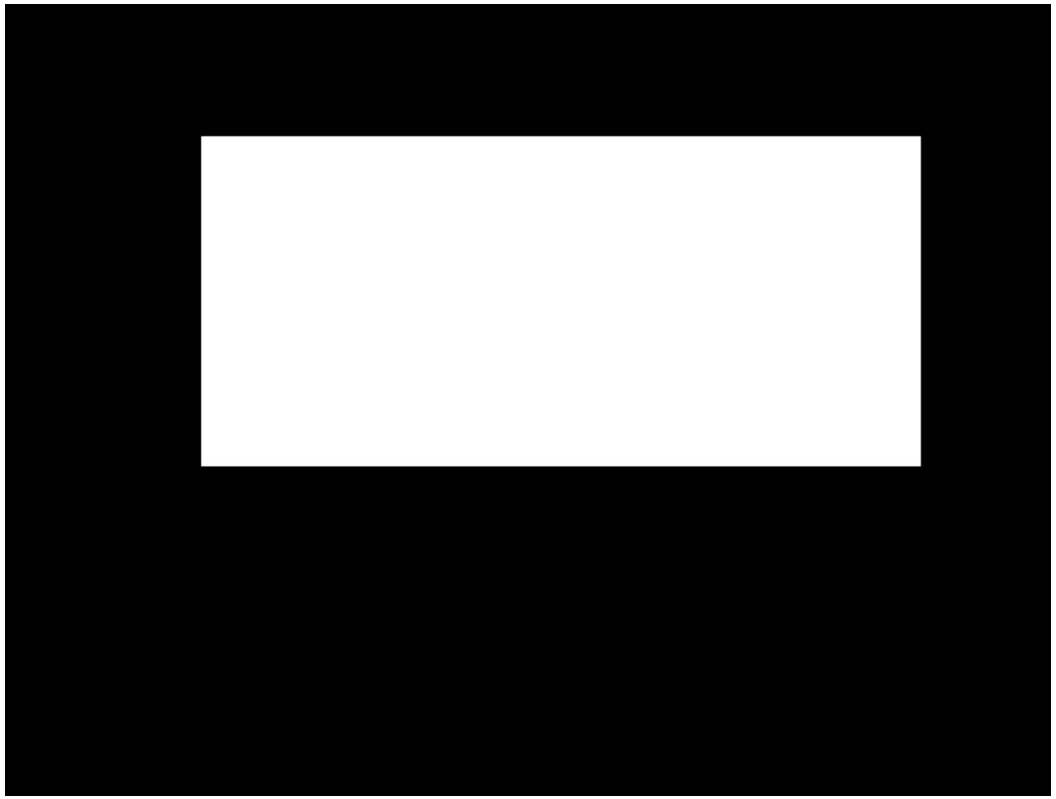
Preprocess

Detection 범위 조정

03. 구현

Preprocess

Detection 범위 조정



03. 구현

Preprocess

Detection 범위 조정

03. 구현

Preprocess

Gamma 조정

03. 구현

Preprocess

Gamma 조정


```
hyunho@hyunho-13ZD940-GX50K:~/Desktop$ alpr -c kr 2.jpg
plate0: 10 results
    - 15우7984   confidence: 94.8451
    - 15오7984   confidence: 81.3401
    - 15무7984   confidence: 81.3373
    - 15구7984   confidence: 81.3315
    - 1597984    confidence: 81.3275
    - 경5우7984  confidence: 81.2941
    - 경5오7984  confidence: 67.7891
    - 경5무7984  confidence: 67.7863
    - 경5구7984  confidence: 67.7805
    - 경597984   confidence: 67.7764
hyunho@hyunho-13ZD940-GX50K:~/Desktop$
```

03

Source Code

03. 구현

Source Code

test_parking.py

```
rom openalpr import Alpr
mport time
mport os
mport csv
mport cv2
mport numpy as np

ef main():
   alpr = Alpr("parking1")

   if not alpr.is_loaded():
       print("Error loading OpenALPR")
   else:
       print("Using OpenALPR " + alpr.get_version())

       # alpr setting
       alpr.set_top_n(1)
       alpr.set_default_region("wa")
       alpr.set_detect_region(False)

       # read file path
       filenames = []
       for fold in os.listdir('parking'):
           for fname in os.listdir(os.path.join('parking', fold)):
               if fname.endswith('.jpg'):
                   filenames.append(os.path.join('parking', fold, fname))
       filenames = sorted(filenames)

       # num of file
       count = len(filenames)
       sum_pt = 0

       # open csv
       csvfile = open('parking.csv', 'w', newline='')
       csvwriter = csv.writer(csvfile, delimiter=',')

       for idx, filename in enumerate(filenames):
           print('(', idx + 1, '/', count, ') Image name: {}'.format(filename))

           # first read image
           original = cv2.imread(filename)

           start = time.time()

           # adjust gamma and write image and reread image
           adjusted = adjust_gamma(original, gamma = 2.0)
           cv2.imwrite(filename, adjusted)
           img = open(filename, "rb").read()

           # recognize image
           results = alpr.recognize_array(img)
           if len(results['results']):
              label = results['results'][0]['plate']
              box = [1600, 1200, 0, 0]
              for result in results['results'][0]['coordinates']:
                 if box[0] > result['x']:
                    box[0] = result['x']
                 if box[1] > result['y']:
                    box[1] = result['y']
                 if box[2] < result['x']:
                    box[2] = result['x']
                 if box[3] < result['y']:
                    box[3] = result['y']
              csvwriter.writerow([filename, label, box[0] + 7, box[1], box[2] - 2, box[3] - 4])
           else:
              csvwriter.writerow([filename, '', 0, 0, 0, 0])

           end = time.time()

           # rewrite original image
           cv2.imwrite(filename, original)

           pt = end - start
           sum_pt += pt
           print("process time :", pt)

       print("average processing time :", float(sum_pt / count * 1000))

 adjust gamma
ef adjust_gamma(image, gamma=1.0):
  invGamma = 1.0 / gamma
  table = np.array([((i / 255.0) ** invGamma) * 255
     for i in np.arange(0, 256)]).astype("uint8")
  return cv2.LUT(image, table)

f __name__ == '__main__':
   main()
```

03. 구현

Source Code

test_parking.py

```
from openalpr import Alpr
import time
import os
import csv
import cv2
import numpy as np

def main():
    alpr = Alpr("parking1")

    if not alpr.is_loaded():
        print("Error loading OpenALPR")
    else:
        print("Using OpenALPR " + alpr.get_version())

        # alpr setting
        alpr.set_top_n(1)
        alpr.set_default_region("wa")
        alpr.set_detect_region(False)

        # read file path
        filenames = []
        for fold in os.listdir('parking'):
            for fname in os.listdir(os.path.join('parking', fold)):
                if fname.endswith('.jpg'):
                    filenames.append(os.path.join('parking', fold, fname))
        filenames = sorted(filenames)

        # num of file
        count = len(filenames)
        sum_pt = 0

        # open csv
        csvfile = open('parking.csv', 'w', newline='')
        csvwriter = csv.writer(csvfile, delimiter=',')
```

03. 구현

Source Code

test_parking.py

```
for idx, filename in enumerate(filenames):
    print('(', idx + 1, '/', count, ') Image name: {}'.format(filename))

    # first read image
    original = cv2.imread(filename)

    start = time.time()

    # adjust gamma and write image and reread image
    adjusted = adjust_gamma(original, gamma = 2.0)
    cv2.imwrite(filename, adjusted)
    img = open(filename, "rb").read()

    # recognize image
    results = alpr.recognize_array(img)
    if len(results['results']):
       label = results['results'][0]['plate']
       box = [1600, 1200, 0, 0]
       for result in results['results'][0]['coordinates']:
          if box[0] > result['x']:
             box[0] = result['x']
          if box[1] > result['y']:
             box[1] = result['y']
          if box[2] < result['x']:
             box[2] = result['x']
          if box[3] < result['y']:
             box[3] = result['y']
       csvwriter.writerow([filename, label, box[0] + 7, box[1], box[2] - 2, box[3] - 4])
    else:
       csvwriter.writerow([filename, '', 0, 0, 0, 0])

    end = time.time()

    # rewrite original image
    cv2.imwrite(filename, original)

    pt = end - start
    sum_pt += pt
    print("process time :", pt)

print("average processing time :", float(sum_pt / count * 1000))
```

03. 구현

Source Code

test_parking.py

```
# adjust gamma
def adjust_gamma(image, gamma=1.0):
   invGamma = 1.0 / gamma
   table = np.array([((i / 255.0) ** invGamma) * 255
      for i in np.arange(0, 256)]).astype("uint8")
   return cv2.LUT(image, table)

if __name__ == '__main__':
    main()
```

03. 구현

Source Code

test_cctv.py

```python
from openalpr import Alpr
import time
import os
import csv

def main():
    alpr = Alpr("cctv1")

    if not alpr.is_loaded():
        print("Error loading OpenALPR")
    else:
        print("Using OpenALPR " + alpr.get_version())

        # alpr setting
        alpr.set_top_n(1)
        alpr.set_default_region("wa")
        alpr.set_detect_region(False)

        # read file path
        filenames = []
        for fold in os.listdir('cctv'):
            for fname in os.listdir(os.path.join('cctv', fold)):
                if fname.endswith('.png'):
                    filenames.append(os.path.join('cctv', fold, fname))
        filenames = sorted(filenames)

        # num of file
        count = len(filenames)
        sum_pt = 0

        # open csv
        csvfile = open('cctv.csv', 'w', newline='')
        csvwriter = csv.writer(csvfile, delimiter=',')

        for idx, filename in enumerate(filenames):
            print('(', idx + 1, '/', count, ') Image name: {}'.format(filename))

            # read image
            img = open(filename, "rb").read()

            # recognize image
            start = time.time()
            results = alpr.recognize_array(img)

            # if find plate
            if len(results['results']):
                label = ''
                box = [1920, 1080, 0, 0]
                rec_size = 0
                # all of finded plate
                for i in range(len(results['results'])):
                    # only first time
                    if i == 0:
                        label = results['results'][i]['plate']
                        for result in results['results'][i]['coordinates']:
                            if box[0] > result['x']:
                                box[0] = result['x']
                            if box[1] > result['y']:
                                box[1] = result['y']
                            if box[2] < result['x']:
                                box[2] = result['x']
                            if box[3] < result['y']:
                                box[3] = result['y']
                        rec_size = (box[3] - box[1]) * (box[2] - box[0])
                    else:
                        temp_box = [1920, 1080, 0, 0]
                        for result in results['results'][i]['coordinates']:
                            if temp_box[0] > result['x']:
                                temp_box[0] = result['x']
                            if temp_box[1] > result['y']:
                                temp_box[1] = result['y']
                            if temp_box[2] < result['x']:
                                temp_box[2] = result['x']
                            if temp_box[3] < result['y']:
                                temp_box[3] = result['y']
                            temp_rec_size = (temp_box[3] - temp_box[1]) * (temp_box[2] - temp_box[0])
                            if rec_size < temp_rec_size:
                                label = results['results'][i]['plate']
                                box[0] = temp_box[0]
                                box[1] = temp_box[1]
                                box[2] = temp_box[2]
                                box[3] = temp_box[3]
                                rec_size = temp_rec_size
                    csvwriter.writerow([filename, label, box[0] + 6, box[1] + 3, box[2] - 3, box[3] - 3])
                else:
                    csvwriter.writerow([filename, '', 0, 0, 0, 0])

                end = time.time()

                pt = end - start
                sum_pt += pt

                print("process time :", pt)

            print ("average processing time :", float(sum_pt / count * 1000))

if __name__ == '__main__':
    main()
```

03. 구현

Source Code

test_cctv.py

```
from openalpr import Alpr
import time
import os
import csv

def main():
    alpr = Alpr("cctv1")

    if not alpr.is_loaded():
        print("Error loading OpenALPR")
    else:
        print("Using OpenALPR " + alpr.get_version())

        # alpr setting
        alpr.set_top_n(1)
        alpr.set_default_region("wa")
        alpr.set_detect_region(False)

        # read file path
        filenames = []
        for fold in os.listdir('cctv'):
            for fname in os.listdir(os.path.join('cctv', fold)):
                if fname.endswith('.png'):
                    filenames.append(os.path.join('cctv', fold, fname))
        filenames = sorted(filenames)

        # num of file
        count = len(filenames)
        sum_pt = 0

        # open csv
        csvfile = open('cctv.csv', 'w', newline='')
        csvwriter = csv.writer(csvfile, delimiter=',')
```

03. 구현

Source Code

test_cctv.py

```
for idx, filename in enumerate(filenames):
    print('(', idx + 1, '/', count, ') Image name: {}'.format(filename))

    # read image
    img = open(filename, "rb").read()

    # recognize image
    start = time.time()
    results = alpr.recognize_array(img)

# if find plate
if len(results['results']):
    label = ''
    box = [1920, 1080, 0, 0]
    rec_size = 0
    # all of finded plate
    for i in range(len(results['results'])):
        # only first time
        if i == 0:
            label = results['results'][i]['plate']
            for result in results['results'][i]['coordinates']:
                if box[0] > result['x']:
                    box[0] = result['x']
                if box[1] > result['y']:
                    box[1] = result['y']
                if box[2] < result['x']:
                    box[2] = result['x']
                if box[3] < result['y']:
                    box[3] = result['y']
            rec_size = (box[3] - box[1]) * (box[2] - box[0])
        else:
            temp_box = [1920, 1080, 0, 0]
            for result in results['results'][i]['coordinates']:
                if temp_box[0] > result['x']:
                    temp_box[0] = result['x']
                if temp_box[1] > result['y']:
                    temp_box[1] = result['y']
                if temp_box[2] < result['x']:
                    temp_box[2] = result['x']
                if temp_box[3] < result['y']:
                    temp_box[3] = result['y']
                temp_rec_size = (temp_box[3] - temp_box[1]) * (temp_box[2] - temp_box[0])
            if rec_size < temp_rec_size:
                label = results['results'][i]['plate']
                box[0] = temp_box[0]
                box[1] = temp_box[1]
                box[2] = temp_box[2]
                box[3] = temp_box[3]
                rec_size = temp_rec_size
    csvwriter.writerow([filename, label, box[0] + 6, box[1] + 3, box[2] - 3, box[3] - 3])
else:
    csvwriter.writerow([filename, '', 0, 0, 0, 0])
```

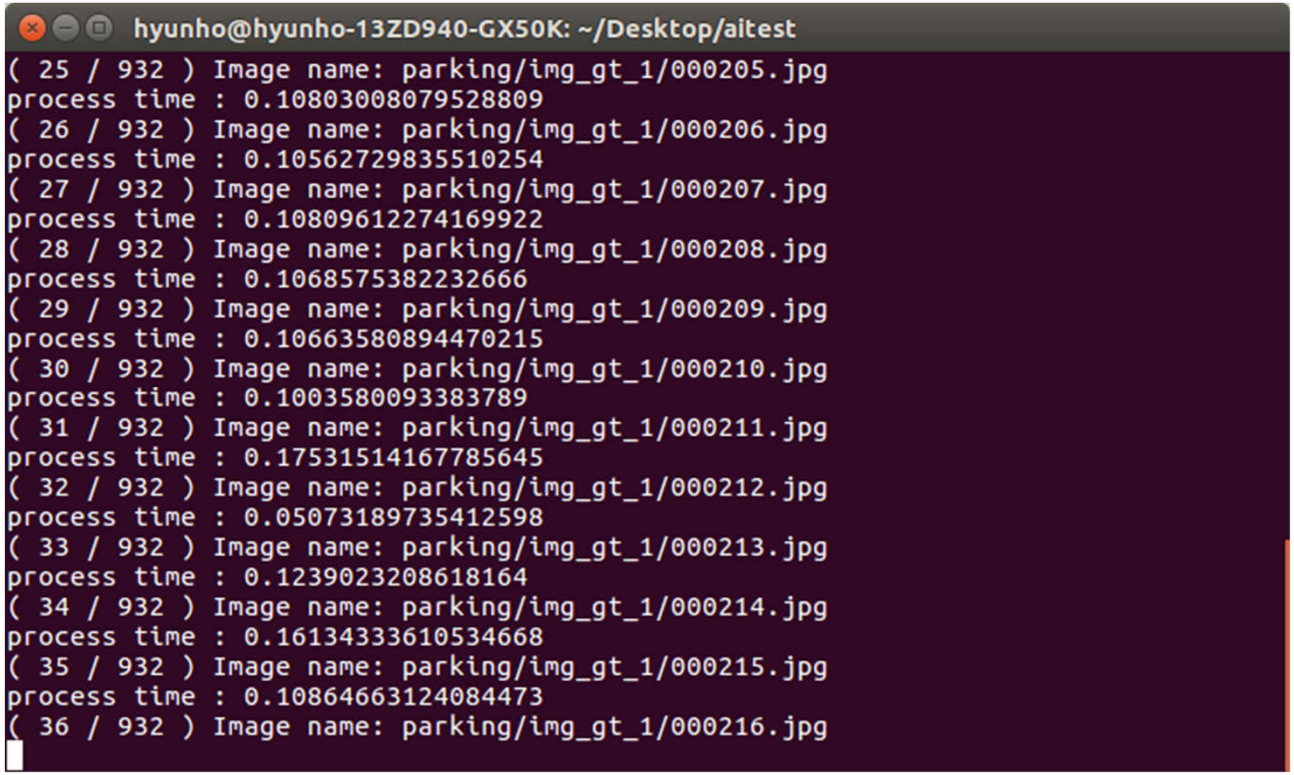
03

Test Result

03. 구현

Test Result

실행 화면



03. 구현

Test Result

측정 결과

Parking data

num_bbox_examples	285
num_bbox_corrects	259
bbox_accuracy	90.88
num_rec_examples	285
num_rec_corrects	258
rec_accuracy	90.53
avg_pt	91.48
score	182.26

CCTV data

num_bbox_examples	451
num_bbox_corrects	349
bbox_accuracy	77.38
num_rec_examples	436
num_rec_corrects	346
rec_accuracy	79.36
avg_pt	76.81
score	159.06

Total Score
344.49

목차

04

Conclusion

04. 결론

Conclusion

- Parking의 경우 detection 범위를 줄였을 경우의 score가 더 높았다.
  이 것은 accuracy의 차이는 별로 없었지만, processing time이 감소하였기 때문이다.

- CCTV의 경우 detection 범위를 줄였을 경우의 score가 더 낮았다.
  이 것은 accuracy가 감소하였기 때문이다.

- 어두운 image의 경우 Gamma 조정을 하게 되면 굉장히 좋은 성능 향상이 있었다.

- Bounding Box를 구할 때, train data의 평균값에 해당하는 거리를 좌표이동을
  시켜 준 경우 좋은 성능 향상이 있었다.

Thank you

감사합니다.

12141755 임현호
12141725 안수진