Assignment 4

Computer Engineering
Hacettepe University
Ankara, Turkey
Melih SUNMAN
b21827809@cs.hacettepe.edu.tr

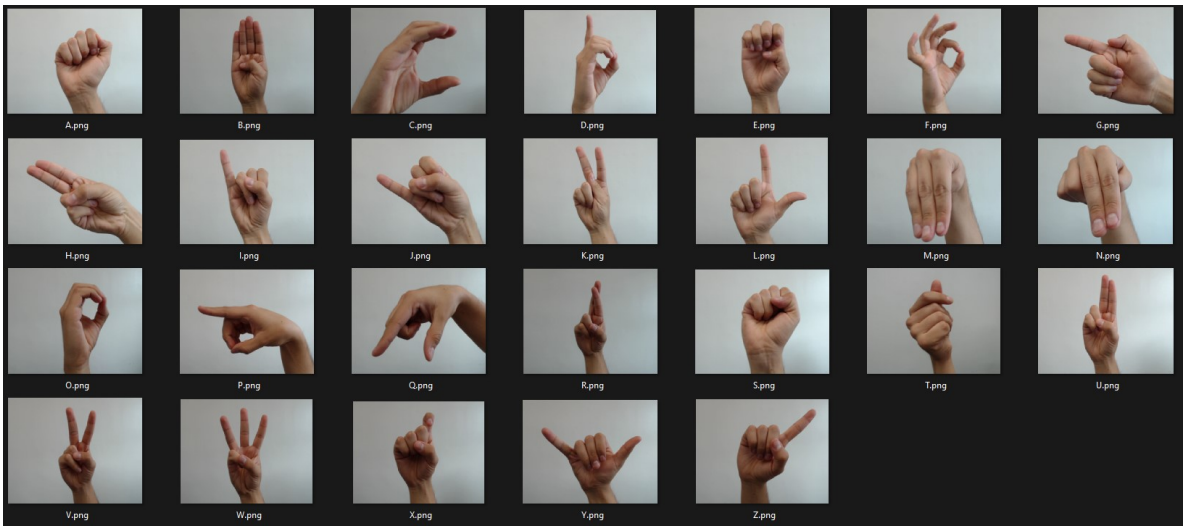
January 7, 2023

1 Introduction

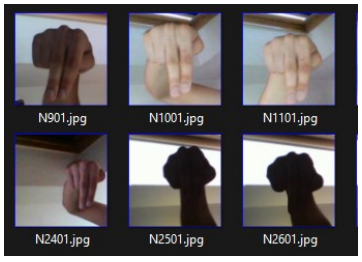
In this project, we will do pattern matching using image processing techniques. Our aim is to identify the letters in the target file by introducing the hand signs alphabet given as an templates to our algorithm.

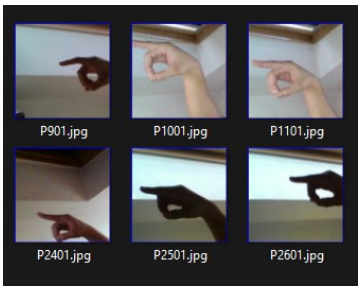
2 Inputs

2.1 Templates



2.2 Targets





3 Code Part

3.1 Importing Necessary Libs

```python
import numpy as np
import argparse
import imutils
import glob
import cv2
import os
```

3.2 Defining the Algorithm Function

```python
def image_finder(template, target, target_dir = 'dataset/target'):
    template = cv2.imread(template)
    template = cv2.cvtColor(template, cv2.COLOR_BGR2GRAY)
    template = cv2.Canny(template, 50, 200)
    (template_height, template_weight) = template.shape[:2]
    cv2.imshow("Template", template)
    all_images = len(target[1])
    success = 0
    for imagePath in target[1]:
        image = cv2.imread(target_dir+'/'+target[0]+'/'+ imagePath)
        gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
        found = None
        for scale in np.linspace(0.2, 1.0, 20)[::-1]:
            resized = imutils.resize(gray, width = int(gray.shape[1] * scale))
            r = gray.shape[1] / float(resized.shape[1])
            if resized.shape[0] < template_height or resized.shape[1] < template_weight:
                break
            edged = cv2.Canny(resized, 50, 50)
            result = cv2.matchTemplate(edged, template, cv2.TM_CCOEFF)
            (_, maxVal, _, maxLoc) = cv2.minMaxLoc(result)
            if found is None or maxVal > found[0]:
                found = (maxVal, maxLoc, r)
        success += 1
        (_, maxLoc, r) = found
        (startX, startY) = (int(maxLoc[0] * r), int(maxLoc[1] * r))
        (endX, endY) = (int((maxLoc[0] + template_weight) * r), int((maxLoc[1] + template_height) * r))
        cv2.rectangle(image, (startX, startY), (endX, endY), (0, 0, 255), 2)
        cv2.imshow(target[0], image)
        cv2.waitKey(0)

    print('Accuricy of',target[0], 'is', (success/all_images), success, all_images)
```

3.3 Reading Datas

```python
temp_dir = 'dataset/template/'
temp_files = os.listdir(temp_dir)

target_dir = 'dataset/target'
target_files = os.listdir(target_dir)

temp_list = []
for i in temp_files:
    temp_list.append(temp_dir+ i)

target_dict = {}
for i in target_files:
    target_dict[i] = (os.listdir(target_dir +'/' +i))

for i in range(len(temp_list)):
    image = temp_list[i]
    target = list(target_dict.items())[i]
    image_finder(image, target)
```

4 Outputs

4.1 Letter A

4.2 Letter D

4.3 Letter F

5 Concludion

- When we look at the results, the defined algorithm mostly detected the target images. Although some target images had errors such as choosing an arm instead of a hand, it worked successfully. With this approach, we achieved an undeniable, if not perfect, result. Advanced neural network techniques can be used for better results.