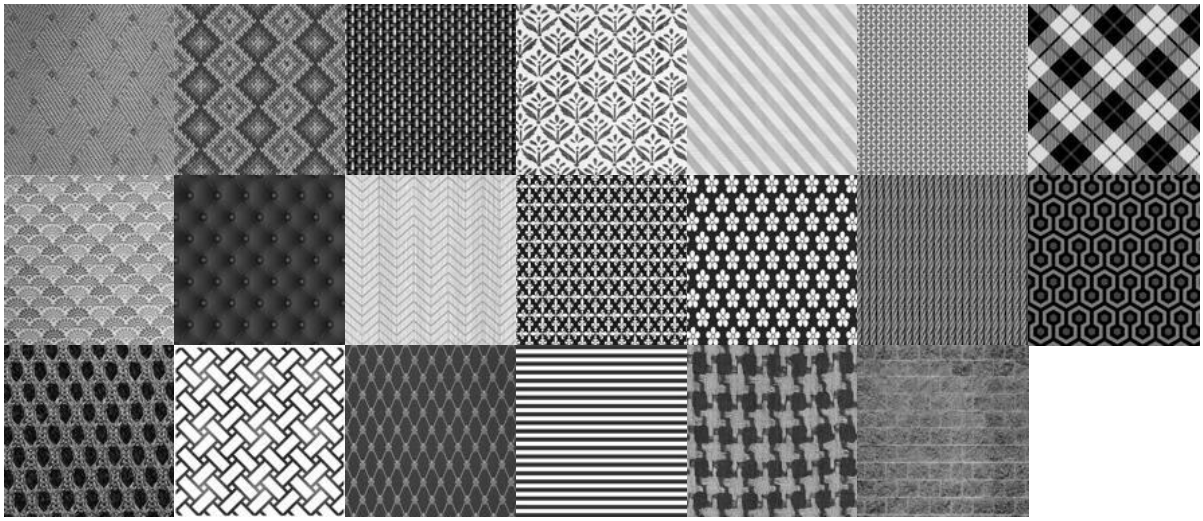
HW#10 DFT

2017029970 우원진

Fabric Patterns



위의 20개의 데이터를 가지고 과제를 진행했습니다. 모든 패턴들의 64 x 64 블록이 패턴을 포함하도록 만들다 보니 크기를 128 x 128로 만들었습니다.

```
# GRAYSCALE 이미지 만들기
# image_set = []
# for i in range(1, 21):
#     image_set.append(str(i) + ".jpg")

# for image in image_set:
#     img = cv2.imread("./"+image, cv2.IMREAD_GRAYSCALE)
#     # img_gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
#     cv2.imwrite("./gray_" + image, img)

# center = int(length/2)
# img = Image.open("gray_20.jpg")
# img = img.resize((128, 128))
# area = (32, 32, 96, 96)
# cropped_img = img.crop(area)
# img.save("20_size.jpg")
# img.show()
# cropped_img.show()
```

이 코드가 위의 사진을 만든 코드입니다.

```
def largest_indices(ary, n):
    """Returns the n largest indices from a numpy array."""
    flat = ary.flatten()
    for i in range(len(flat)):
        flat[i] = abs(flat[i])
    indices = np.argpartition(flat, -n)[-n:]
    indices = indices[np.argsort(-flat[indices])]
    return np.unravel_index(indices, ary.shape)
```

이 함수는 Magnitude Matrix를 구했을때 거기서 가장 큰 값에 해당하는 index를 array로 반환해주는 함수입니다.

```
largest_indices_with_patterns = []
for i in range(1, 21):
    img = cv2.imread("./"+str(i)+"_size.jpg", 0)
    m_sum = np.zeros((64, 64))
    for j in range(10):
        r_row = random.randint(0, 64)
        r_col = random.randint(0, 64)
        img_crop = img[r_row:r_row+64, r_col:r_col+64]
        # print(img_crop)
        f = np.fft.fft2(img_crop)
        fshift = np.fft.fftshift(f)
        magnitude = np.abs(fshift)
        m_sum += magnitude
    m_average = m_sum/10
    row_arr, col_arr = largest_indices(m_average, 11)
    temp_list = []
    for k in range(11):
        temp_list.append((row_arr[k], col_arr[k]))
    largest_indices_with_patterns.append(temp_list)
# Average Magnitude 완성
```

이 부분은 20개의 사진에 대해 각각 10번씩 random하게 Block을 바꿔가며 Average를 구해준뒤 그 사진의 평균에 대한 가장 큰 10개의 index를 largest_indices_with_patterns 라는 변수에 list형태로 담았습니다. 그런데 10개를 구해야 하지만 DC성분은 빼주기 위해 11개를 뽑았고 나중에 DC성분을 제외할 생각입니다.

그리고 임의의 사진에 대해 Pattern Recognition하는 함수입니다.

```
# statistic의 #1 은 사진 번호, #2 는 count sum, #3은 random하게 나온횟수
for k in range(1000):
    pattern_num = random.randint(1, 20)
    # print("pattern : ", pattern_num)
    pattern_img = cv2.imread("./"+str(pattern_num)+"_size.jpg", 0)
    random_row = random.randint(0, 64)
    random_col = random.randint(0, 64)
    pattern_block = pattern_img[random_row:random_row +
                                64, random_col:random_col+64]
    f = np.fft.fft2(pattern_block)
    fshift = np.fft.fftshift(f)
    magnitude = np.abs(fshift)
    arr1, arr2 = largest_indices(magnitude, 10)
    largest_indices_result = []
    for i in range(10):
        largest_indices_result.append((arr1[i], arr2[i]))

    result = (-1, -1)
    for i in range(20):
        indices = largest_indices_with_patterns[i]
        count = 0
        for (row, col) in largest_indices_result:
            if (row, col) in indices:
                count += 1
        (idx, cnt) = result
        if count > cnt:
            result = (i, count)
    if pattern_num != result[0]+1:
        print("Recognition Error !!")
    else:
        statistics[result[0]][1] += result[1]-1
        statistics[result[0]][2] += 1
        # print("계산결과 pattern : ", result[0]+1, result[1]-1)
print(statistics)
for i in range(20):
    print(str(i+1) + "번째 사진은 1000번중 랜덤하게 " +
          str(statistics[i][2]) + "번 나왔으며 가장 큰 coefficients 10개 중 평균 " + str(statistics[i][1]/statistics[i
```

실행 결과

```
1번째 사진은 1000번중 랜덤하게 54번 나왔으며 가장 큰 coefficients 10개 중 평균 7.2037037037037037개가 똑같다.
2번째 사진은 1000번중 랜덤하게 44번 나왔으며 가장 큰 coefficients 10개 중 평균 7.909090909090909개가 똑같다.
3번째 사진은 1000번중 랜덤하게 47번 나왔으며 가장 큰 coefficients 10개 중 평균 9.0개가 똑같다.
4번째 사진은 1000번중 랜덤하게 47번 나왔으며 가장 큰 coefficients 10개 중 평균 7.5744680851063830개가 똑같다.
5번째 사진은 1000번중 랜덤하게 47번 나왔으며 가장 큰 coefficients 10개 중 평균 9.0개가 똑같다.
6번째 사진은 1000번중 랜덤하게 55번 나왔으며 가장 큰 coefficients 10개 중 평균 9.0개가 똑같다.
7번째 사진은 1000번중 랜덤하게 60번 나왔으며 가장 큰 coefficients 10개 중 평균 8.5개가 똑같다.
8번째 사진은 1000번중 랜덤하게 60번 나왔으며 가장 큰 coefficients 10개 중 평균 7.7833333333333333개가 똑같다.
9번째 사진은 1000번중 랜덤하게 52번 나왔으며 가장 큰 coefficients 10개 중 평균 7.9807692307692310개가 똑같다.
10번째 사진은 1000번중 랜덤하게 32번 나왔으며 가장 큰 coefficients 10개 중 평균 5.5625개가 똑같다.
11번째 사진은 1000번중 랜덤하게 46번 나왔으며 가장 큰 coefficients 10개 중 평균 8.5217391304347830개가 똑같다.
12번째 사진은 1000번중 랜덤하게 44번 나왔으며 가장 큰 coefficients 10개 중 평균 8.1818181818181820개가 똑같다.
13번째 사진은 1000번중 랜덤하게 50번 나왔으며 가장 큰 coefficients 10개 중 평균 8.9개가 똑같다.
14번째 사진은 1000번중 랜덤하게 44번 나왔으며 가장 큰 coefficients 10개 중 평균 8.3181818181818180개가 똑같다.
15번째 사진은 1000번중 랜덤하게 53번 나왔으며 가장 큰 coefficients 10개 중 평균 9.0개가 똑같다.
16번째 사진은 1000번중 랜덤하게 49번 나왔으며 가장 큰 coefficients 10개 중 평균 8.2448979591836730개가 똑같다.
17번째 사진은 1000번중 랜덤하게 53번 나왔으며 가장 큰 coefficients 10개 중 평균 8.1509433962264150개가 똑같다.
18번째 사진은 1000번중 랜덤하게 59번 나왔으며 가장 큰 coefficients 10개 중 평균 8.9152542372881350개가 똑같다.
19번째 사진은 1000번중 랜덤하게 54번 나왔으며 가장 큰 coefficients 10개 중 평균 8.6666666666666660개가 똑같다.
20번째 사진은 1000번중 랜덤하게 50번 나왔으며 가장 큰 coefficients 10개 중 평균 7.6개가 똑같다.
```

위의 코드에서 “Recognition Error!! ”부분을 출력하는 부분이 한번도 실행되지 않은 보아 Pattern 인식이 잘 되는 것 같습니다. 그리고 각각의 사진마다 10개의 Coefficient 중 같은것의 갯수를 count 했고 평균적으로 몇개 정도가 같은지 분석했습니다.