Tugas Praktikum Minggu 3

Nicholas Juan Kalvin P. | 162012133068

Image Mining

Tugas Praktikum:

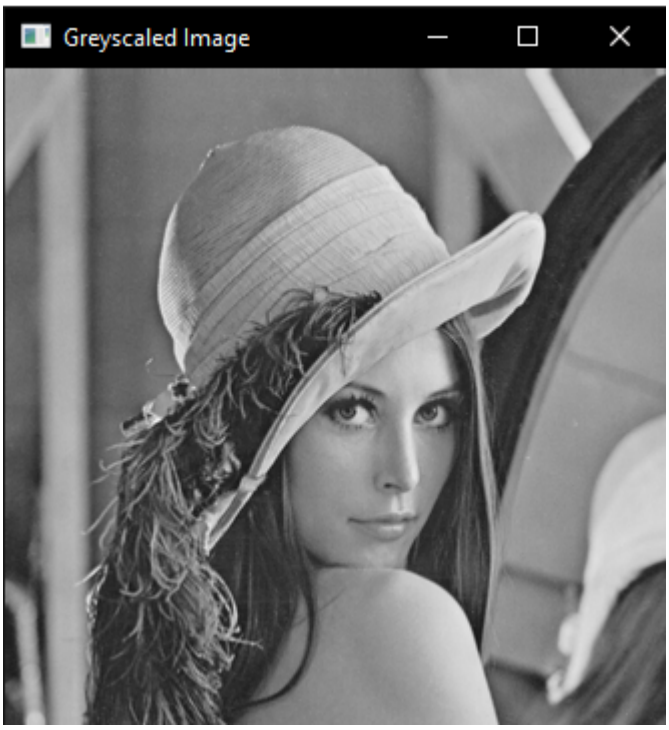
1. Memodifikasi kode bagian (a) agar dapat menampilkan citra "Lenna" dalam grayscale
2. Memodifikasi kode bagian (a) agar bisa melakukkan crop pada citra "Lenna"
3. Memodifikasi kode bagian (a) agar dapat menampilkan ukuran citra grayscale dar "Lenna" dan nilai matriks dari citra grayscale pada baris ke-0 dan kolom ke-0. Apakah hasilnya beda dari kode bagian (b)? Jelaskan alasannya!
4. Modifikasi kode bagian (c) untuk menampilkan Green and Red
5. Simpanlah file "Lenna" menjadi format JPEG dengan menggunakan method `imwrite` pada OpenCV. Apakah terdapat perbedaan nilai array pada file citra asli dan file dengan format JPEG? Jelaskan alasannya!

In [ ]:
```
# Modules
import sklearn
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import OneHotEncoder
from scipy.io.wavfile import write
import matplotlib.pyplot as plt
import numpy as np
import random
import cv2
```

Tugas Praktikum No. A 1

In [ ]:
```
# Load image
normal_image = cv2.imread('./lenna.png')
grey_image = cv2.cvtColor(normal_image, cv2.COLOR_BGR2GRAY)
cv2.imshow('Greyscaled Image', grey_image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Gambar yang dihasilkan adalah



Tugas Praktikum No. A 2

In [ ]:

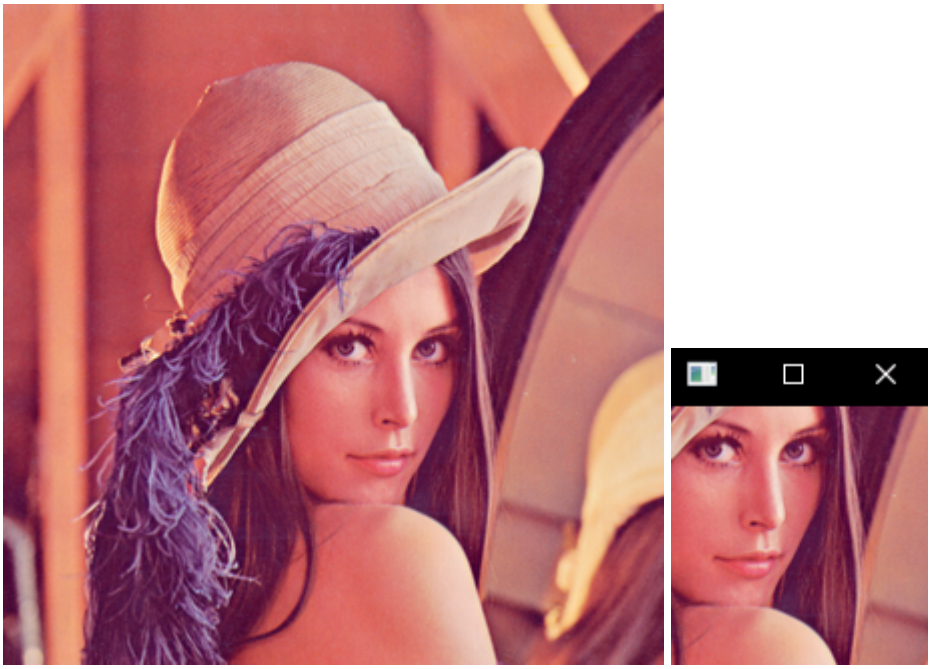
```
# Mencari dimensi
height = normal_image.shape[0]
width = normal_image.shape[1]
channels = normal_image.shape[2]

print('Height: ', height)
print('Width: ', width)
print('Channels: ', channels)

# Croppping
cropped_image = normal_image[150:280, 150:280]
cv2.imshow('Cropped Image', cropped_image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

```
Height:  330
Width:  330
Channels:  3
```

Gambar hasil crop terlihat seperti berikut:



Tugas Praktikum No. A 3

In [ ]:
```
print('Ukuran citra greyscale: ', grey_image.shape)
print('Matriks warna (0, 0) greyscale:  ', grey_image[0, 0], '\n')

print('Ukuran citra warna: ', normal_image.shape)
print('Matriks warna (0, 0) greyscale:  ', normal_image[0, 0])
```

```
Ukuran citra greyscale:  (330, 330)
Matriks warna (0, 0) greyscale:   162

Ukuran citra warna:  (330, 330, 3)
Matriks warna (0, 0) greyscale:   [124 137 226]
```

Penjelasan:

Perbedaan pada nilai ukuran citra dan matriks warna terdapat dijelaskan oleh warna citra itu sendiri. Pada ukuran citra greyscale, hanya terdapat 2 nilai karena nilai 3 yang berisi `channel` tidak tersedia karena hanya memiliki satu spektrum warna.

Matriks pada citra greyscale juga hanya berjumlah satu, karena satu nilai itu yang menjelaskan seberapa "dalam" warna hitam yang dibuat

Sedangkan untuk citra yang memiliki warna, terdapat nilai channel bernilai 3 yaitu channel `Red, Green dan Blue` yang membuat citra memiliki warna.

Matriks pada citra bewarna juga berbeda, karena matriks ini memiliki 3 nilai, yaitu `Red, Green dan Blue` dimana masing nilai menjelaskan seberapa "dalam" warna tersebut

Tugas Praktikum No. A 4

In [ ]:
```
# Lihat Channel Warna
(blue, green, red) = cv2.split(normal_image)
print('Channel Biru: ', blue)
print('Channel Hijau: ', green)
print('Channel Merah: ', red)

def single_colorizer(image, color):
    splitted_image = cv2.split(image)
    m = np.zeros(splitted_image[0].shape[:2], splitted_image[0].dtype)
    colored_image = None
    if color == 'red':
        colored_image = cv2.merge([m, m, splitted_image[0]])
    elif color == 'green':
        colored_image = cv2.merge([m, splitted_image[0], m])
    elif color == 'blue':
        colored_image = cv2.merge([splitted_image[0], m, m])
    else:
        print('Error!')

    return colored_image

# cv2.imshow('biru', single_colorizer(blue, 2))
red = single_colorizer(normal_image, 'red')
green = single_colorizer(normal_image, 'green')
blue = single_colorizer(normal_image, 'blue')

vertical_concat = np.concatenate((red, green, blue), axis=1)
cv2.imshow("Citra setiap channel", vertical_concat)
cv2.waitKey(0)
```

```
Channel Biru:  [[124 131 125 ... 124 120  96]
 [124 131 125 ... 124 119  96]
 [125 132 126 ... 126 122  98]
 ...
 [ 64  60  63 ...  79  77  77]
 [ 59  58  62 ...  83  80  79]
 [ 57  60  64 ...  82  78  82]]
Channel Hijau:  [[137 137 137 ... 149 145 109]
 [137 137 137 ... 148 144 108]
 [137 137 137 ... 151 150 113]
 ...
 [ 27  25  27 ...  64  67  61]
 [ 18  25  28 ...  69  71  67]
 [ 22  29  30 ...  70  69  74]]
Channel Merah:  [[226 224 224 ... 233 229 207]
 [226 224 224 ... 232 229 206]
 [226 223 224 ... 235 233 212]
 ...
 [ 89  88  91 ... 161 168 164]
 [ 82  90  95 ... 172 175 177]
 [ 81  91  96 ... 177 180 184]]
```

Out[ ]: -1

Hasil citra dengan setiap channel warna dipisahkan terlihat seperti berikut:

Tugas Praktikum No. A 5

In [ ]:
```
cv2.imwrite('lenna_jpeg.jpeg', normal_image)
```

Out[ ]: True

In [ ]:
```
png_lenna = cv2.imread("lenna.png")
jpeg_lenna = cv2.imread("lenna_jpeg.jpeg")
```

In [ ]:
```
print(list(set(png_lenna[0][0]) - set(jpeg_lenna[0][0])))
```

```
[226, 124]
```

Dari pengurangan array terhadap array PNG, diketahui terdapat perbedaan pada kedua array dengan index 0,0. Dari sini dapat dibuktikan bahwa PNG dan JPEG memiliki array yang berbeda, karena hasil kompresi

---

Audio Mining

Tugas Praktikum:

1. Modifikasi kode bagian (a) agar membuat gelombang suara dengan frekuensi 1000, dan simpan file audio kedalam file2.wav. Apakah suara yang dihasilkan file2.wav berbeda dengan file1.wav (dihasilkan oleh kode (b))? jelaskan alasannya.
2. Modifikasi kode bagian (a) dengan mengganti nilai amplitudo menjadi 16, dengan frekuensi 200 Hz, lalu simpan file audio kedalam file3.wav. Apakah terdapat perbedaan bunyi dengan hasil suara dari kode bagian (b) atau file1.wav? jelaskan alasannya.

Tugas Praktikum No. B 1

In [ ]:
```
sr = 44100
```

```
freq = 1
length = 1
t = np.arange(0, length, 1.0/sr)
signal = np.sin(np.pi*2*freq*t)


write("file.wav", sr, signal)
```

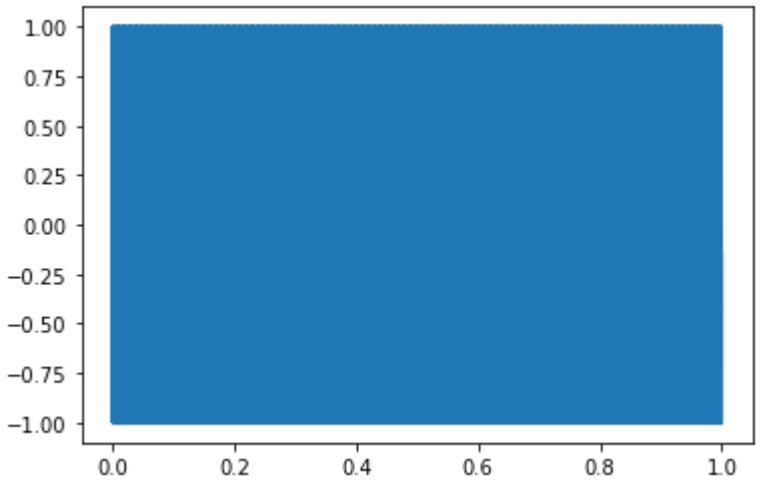
In [ ]:

```
sr = 44100
freq = 1000
length = 1
t = np.arange(0, length, 1.0/sr)
signal = np.sin(np.pi*2*freq*t)


write("file2.wav", sr, signal)
```

In [ ]:

```
plt.plot(t, signal)
plt.show()
```



Gelombang suara yang dihasilkan dengan frekuensi lebih tinggi akan terdengar bernada lebih tinggi yang terdengar pada file2.wav; Sedangkan, frekuensi yang lebih rendah akan mengeluarkan nada yang rendah seperti pada file.wav; Hal ini terjadi sesuai dengan hukum fisika yang berlaku

Tugas Praktikum No. B 2

In [ ]:

```
sr = 44100
freq = 200
length = 1
t2 = np.arange(0, length, 1.0/sr)
signal2 = 16 * np.sin(np.pi*2*freq*t)

write("file3.wav", sr, signal2)
```

Secara saintifik, mengganti amplitudo pada suatu gelombang suara akan menghasilkan suara yang lebih besar. Namun, dalam hal ini, mengganti amplitudo memperbesar suara tetapi volume suara

mungkin dibatasi oleh audio driver dari Python yang akhirnya menghasilkan suara dengan volume yang sama, namun sedikit terdistorsi

In [ ]:

Text Mining

Tugas Praktikum:

1. Modifikasi kode bagian (a) agar bisa menapilkan ASCII code untuk kata $datA \min \in g$
2. Tambahkan kode bagian (b) agar bisa menampilkan kembali kata pertama yang di lakukan one-hot encoding (hint: lakukan inverse dari hasil encode). [[**1**000][0001][0100][0010]] dengan bold adalah kata "I"
3. Download file tugas_text_representation.csv dari hebat, kemudian lakukan CountVectorizer dan TF-IDF pada korpus tersebut. Jelaskan hasil yang didapatkan.
4. Modifikasi kode bagian (d) agar bisa menampilkan grafik dari tiap kata.

Tugas Praktikum No. C 1

In [ ]:
```
_char = 'datA mining'
_ascii = [ord(letter) for letter in _char]
print(_ascii)
```

```
[100, 97, 116, 65, 32, 109, 105, 110, 105, 110, 103]
```

Tugas Praktikum No. C 2

In [ ]:
```
from tkinter import Label


docs = "I ate an apple"

# Memisah kalimat menjadi token
split_docs = docs.split(" ")
data = [doc.split(" ") for doc in split_docs]
values = np.array(data).ravel()
print(values)

# Integer Encoder
label_encoder = LabelEncoder()
integer_encoded = label_encoder.fit_transform(values)
print(integer_encoded)

# Binary Encoder
onehot_encoder = OneHotEncoder(sparse=False)
integer_encoded = integer_encoded.reshape(len(integer_encoded), 1)
onehot_encoded = onehot_encoder.fit_transform(integer_encoded)
```

```
# reverse_onehot = onehot_encoder.inverse_transform(onehot_encoded)
# reverse_encoded = label_encoder.fit_transform(reverse_onehot)

# reverse_encoded
onehot_encoded
```

```
['I' 'ate' 'an' 'apple']
[0 3 1 2]
```

Out[ ]:
```
array([[1., 0., 0., 0.],
       [0., 0., 0., 1.],
       [0., 1., 0., 0.],
       [0., 0., 1., 0.]])
```

In [ ]:
```
def reverse_encoder(encode,cols):
    df = pd.DataFrame(encode, columns=cols)
    df['ind_all'] = (df.iloc[:, :] == 1).idxmax(1)

    return ' '.join([i for i in df['ind_all']])

reverse_encoding = reverse_encoder(encode=onehot_encoded, cols=values)
reverse_encoding
```

Out[ ]:
```
'I apple ate an'
```

Tugas Praktikum No. C 3

In [ ]:
```
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
```

In [ ]:
```
texts = pd.read_csv('tugas_text_representation.csv')
text = [text for text in texts['berita'].tolist()]
print(text)
```

```
['Nlp adalah bagian dari text mining dan banyak aplikasi yang dibuat dengan menggunakan
nlp', 'nlp memiliki hubungan erat dengan proses neurologi yaitu bahasa dan juga pola per
ilaku manusia', 'Analisis sintaksis adalah teknik utama yang digunakan untuk menyelesaik
an tugas nlp']
```

In [ ]:
```
vectorizer = CountVectorizer()

# tokenasi dan membuat vocab
vectorizer.fit(text)
vectorizer_count = vectorizer.vocabulary_
vector = vectorizer.transform(text)

# hasil encode vektor
print(vector.shape)
print(vector.toarray())
```

```
(3, 32)
[[1 0 1 1 0 1 1 1 1 1 0 0 0 0 0 0 1 0 1 0 2 0 0 0 0 0 1 0 0 0 0 1]
 [0 0 0 0 1 0 1 0 1 0 0 1 1 1 1 1 0 0 0 1 1 1 1 1 0 0 0 0 0 0 1 0]
 [1 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 1 0 0 0 1 1 0 1 1 1 0 1]]
```

In [ ]:
```
tf = TfidfVectorizer()
txt_fitted = tf.fit(text)
txt_transformed = txt_fitted.transform(text)

idf = tf.idf_
print(dict(zip(txt_fitted.get_feature_names(), idf)))
```

```
{'adalah': 1.2876820724517808, 'analisis': 1.6931471805599454, 'aplikasi': 1.6931471805599454, 'bagian': 1.6931471805599454, 'bahasa': 1.6931471805599454, 'banyak': 1.6931471805599454, 'dan': 1.2876820724517808, 'dari': 1.6931471805599454, 'dengan': 1.2876820724517808, 'dibuat': 1.6931471805599454, 'digunakan': 1.6931471805599454, 'erat': 1.6931471805599454, 'hubungan': 1.6931471805599454, 'juga': 1.6931471805599454, 'manusia': 1.6931471805599454, 'memiliki': 1.6931471805599454, 'menggunakan': 1.6931471805599454, 'menyelesaikan': 1.6931471805599454, 'mining': 1.6931471805599454, 'neurologi': 1.6931471805599454, 'nlp': 1.0, 'perilaku': 1.6931471805599454, 'pola': 1.6931471805599454, 'proses': 1.6931471805599454, 'sintaksis': 1.6931471805599454, 'teknik': 1.6931471805599454, 'text': 1.6931471805599454, 'tugas': 1.6931471805599454, 'untuk': 1.6931471805599454, 'utama': 1.6931471805599454, 'yaitu': 1.6931471805599454, 'yang': 1.2876820724517808}
```

Tugas Praktikum No. C 4

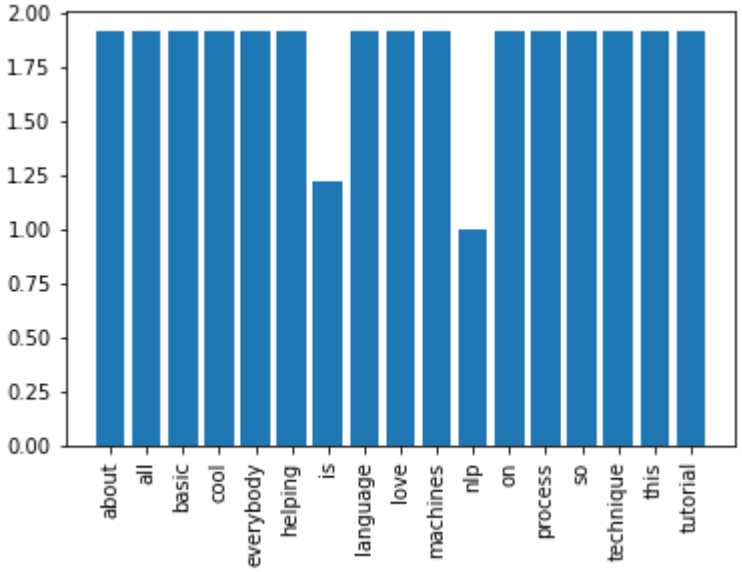
In [ ]:
```
from sklearn.feature_extraction.text import TfidfVectorizer

tf = TfidfVectorizer()
txt_fitted = tf.fit(text)
txt_transformed = txt_fitted.transform(text)

idf = tf.idf_
tfidf_dict = dict(zip(txt_fitted.get_feature_names(), idf))

plt.bar(tfidf_dict.keys(), tfidf_dict.values())
plt.xticks(rotation=90)
```

Out[ ]:
```
([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16],
 [Text(0, 0, ''),
  Text(0, 0, ''),
  Text(0, 0, ''),
  Text(0, 0, ''),
  Text(0, 0, ''),
  Text(0, 0, ''),
  Text(0, 0, ''),
  Text(0, 0, ''),
  Text(0, 0, ''),
  Text(0, 0, ''),
  Text(0, 0, ''),
  Text(0, 0, ''),
  Text(0, 0, ''),
  Text(0, 0, ''),
  Text(0, 0, ''),
  Text(0, 0, ''),
  Text(0, 0, '')])
```


2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
about
all
basic
cool
everybody
helping
is
language
love
machines
nlp
on
process
so
technique
this
tutorial