

TUGAS (Image Processing)

Nama : **Narinda Genta Rosasia**

Nim : **320200401019**

Prodi : **T. Informatika**

Pertemuan 4

Pada pertemuan ini mempelajari beberapa modul library python, seperti matplotlib, PIL, cv2, dan numpy.

Analisis kode pada pengolahan citra menunjukkan beberapa teknik dalam pemrosesan gambar seperti mengubah ukuran gambar, memanggil histogram, dan manipulasi eksposur gambar. Metode ini sangat penting dalam pengolahan data gambar untuk memperbaiki kualitas gambar, memperjelas detail, dan mengoptimalkan representasi data. Menurut Li et al. (2020), teknik pemrosesan gambar seperti histogram equalization dan kontras enhancement dapat meningkatkan deteksi pada aplikasi pengolahan gambar seperti klasifikasi, segmentasi, dan pengenalan objek. Sementara itu, berdasarkan studi yang dilakukan oleh Zhou et al. (2021), teknik rescale intensity dapat membantu meningkatkan ketajaman dan kontras gambar pada sistem penglihatan mesin. Oleh karena itu, pemahaman terhadap teknik-teknik ini sangat penting bagi pengembang yang ingin meningkatkan kualitas gambar dan performa pada aplikasi pengolahan gambar.

Manipulasi gambar dapat menggunakan Python dan library seperti matplotlib, numpy, PIL, dan skimage. Kode-kode tersebut memungkinkan untuk mengimpor gambar ke dalam array numpy, menampilkan gambar dan histogram, mengubah ukuran gambar, mengubah eksposur gambar, serta menampilkan histogram dari gambar yang telah dimanipulasi. Analisis kode-kode tersebut menunjukkan bahwa manipulasi gambar dapat dilakukan dengan mudah menggunakan Python dan library-library yang ada. Hal ini memungkinkan pengguna untuk memodifikasi gambar untuk keperluan yang beragam, seperti pengolahan citra, pemrosesan gambar, atau aplikasi lain yang memerlukan manipulasi gambar.

Berikut link pengerjaan google collab :

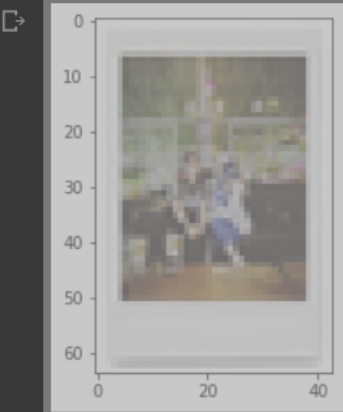
https://colab.research.google.com/drive/1UKsPPPX3CyTqJQnHj_9zeHq-igkKOkMT?usp=sharing

Code #1: Mengubah ukuran gambar

```
#mengubah ukuran gambar contoh op pikse;
import matplotlib.pyplot as plt
from PIL import Image

img= Image.open('/content/drive/MyDrive/Foto/img (1).jpg')
img.thumbnail((64,64), Image.ANTIALIAS)
plt.imshow (img)

plt.show()
```



Kode tersebut merupakan contoh bagaimana mengubah ukuran gambar dengan menggunakan library matplotlib dan PIL (Python Imaging Library). Pada kode tersebut, gambar dibuka menggunakan fungsi `Image.open()` dari PIL kemudian ukuran gambar diubah menggunakan metode `thumbnail()` dengan ukuran 64 x 64 piksel. Hasilnya kemudian ditampilkan dengan menggunakan fungsi `imshow()` dan `plt.show()` dari matplotlib. Dengan menggunakan metode `thumbnail()`, gambar dapat diubah ukurannya secara proporsional tanpa merusak aspek rasio atau proporsi gambar asli.

Code #2: memanggil historigram dengan plt.hist

```
#memanggil histogram plt.hist()

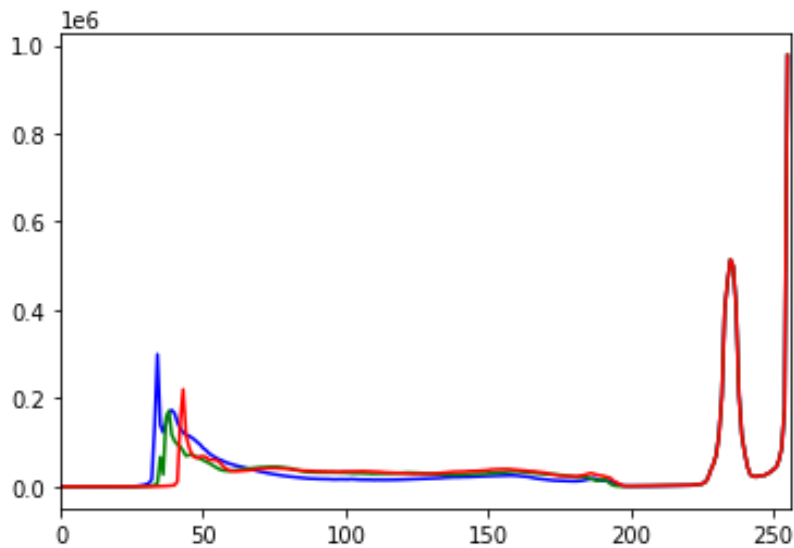
import cv2
import numpy as np

from matplotlib import pyplot as plt

img = cv2.imread('/content/drive/MyDrive/Foto/img (1).jpg')
color = ('b', 'g', 'r')

for i, col in enumerate(color):
    histr = cv2.calcHist([img], [i], None, [256], [0,256])
    plt.plot(histr, color = col)
    plt.xlim([0,256])

plt.show()
```



Kode di atas merupakan contoh bagaimana menghitung dan menampilkan histogram dari gambar menggunakan OpenCV dan Matplotlib di Python. Pertama, gambar kucing diambil menggunakan `cv2.imread()` dan disimpan ke dalam variabel `img`. Selanjutnya, kita menentukan daftar warna yang akan dihitung histogramnya, yaitu biru (b), hijau (g), dan merah (r). Di dalam loop, variabel `histr` akan berisi histogram dari komponen warna yang sedang diperiksa. Kemudian, histogram tersebut ditampilkan menggunakan `plt.plot()`, dengan warna sesuai dengan komponen warna yang sedang diperiksa. Akhirnya, batas sumbu x diatur dari 0 hingga 256, lalu histogram ditampilkan menggunakan `plt.show()`. Histogram ini dapat membantu kita memahami distribusi warna dalam gambar dan berguna dalam proses analisis citra digital.

Code #3: Mengimpor data gambar ke array numpy

```
#Mengimpor data gambar ke dalam array Numpy

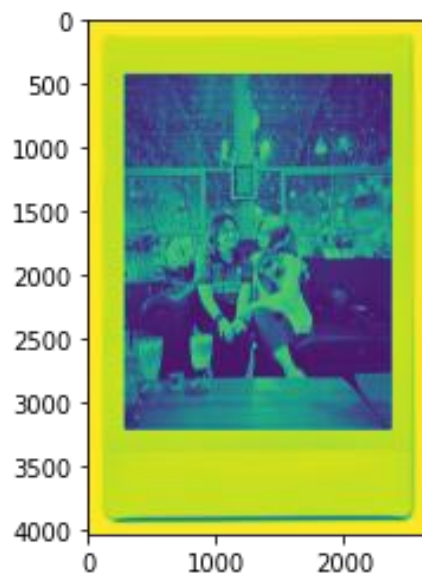
import matplotlib.pyplot as plt
import matplotlib.image as mpimg

from PIL import Image

img = mpimg.imread('/content/drive/MyDrive/Foto/img (1).jpg')
lum_img = img[:, :, 0]

plt.imshow(lum_img)

plt.show()
```



Kode tersebut merupakan contoh pengimporan data gambar ke dalam array NumPy menggunakan modul matplotlib dan PIL. Setelah gambar dibaca menggunakan modul mpimg, gambar diubah ke dalam array NumPy menggunakan indeksasi dan slicing. Kemudian, array tersebut diplot menggunakan metode imshow() dari modul plt untuk menampilkan gambar dalam bentuk 2 dimensi. Potongan kode ini memberikan kemampuan untuk membaca gambar dan memprosesnya menggunakan berbagai teknik pemrosesan gambar, seperti konvolusi, pengelompokan warna, dan analisis citra.

Code #4: Mengimpor dalam imgplot cmap nipy_spectral

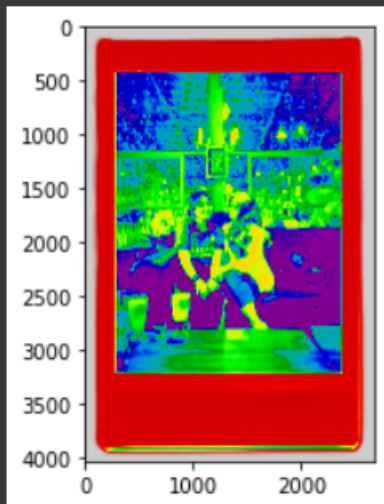
```
#Mengimpor data gambar ke dalam array Numpy

import matplotlib.pyplot as plt
import matplotlib.image as mpimg
from PIL import Image

img = mpimg.imread('/content/drive/MyDrive/Foto/img (1).jpg')
lum_img = img[:, :, 0]

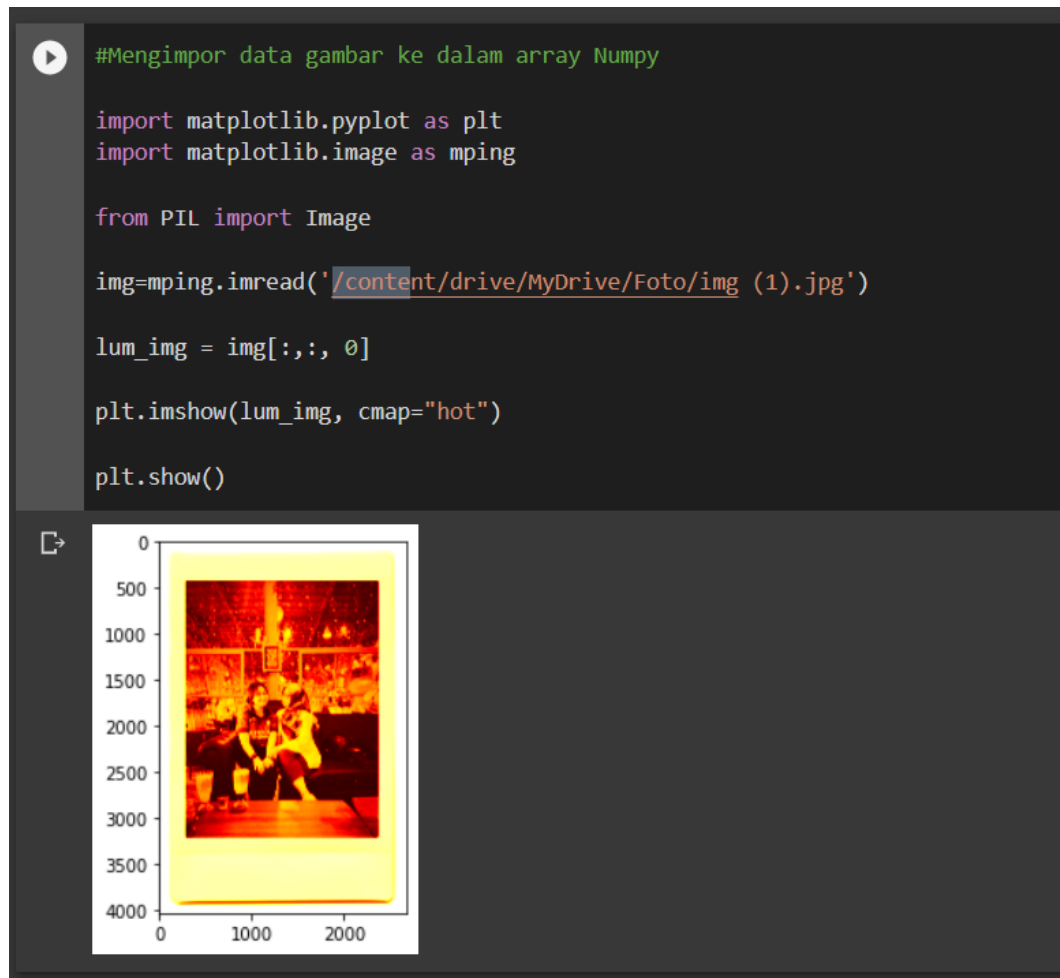
imgplot = plt.imshow(lum_img)
imgplot.set_cmap('nipy_spectral')

plt.show()
```



Kode di atas berfungsi untuk mengimpor data gambar ke dalam array Numpy. Pertama, kita mengimpor modul `matplotlib.pyplot` dan `matplotlib.image` sebagai `mpimg`, dan modul `PIL` sebagai `Image`. Selanjutnya, kita menggunakan fungsi `mpimg.imread()` untuk membaca data gambar dengan format `jpg`. Kemudian, kita memproses data gambar dengan memilih saluran warna ke-0 (warna merah) menggunakan indexing pada array Numpy. Hasilnya disimpan pada variabel `lum_img`. Selanjutnya, kita menggunakan fungsi `plt.imshow()` untuk menampilkan data gambar pada sumbu `x` dan `y`. Fungsi `set_cmap()` digunakan untuk mengatur skema warna yang digunakan dalam tampilan gambar. Terakhir, kita menggunakan fungsi `plt.show()` untuk menampilkan gambar hasil olahan.

Code #5 : Mengimpor dalam imgplot cmap hot



Kode tersebut merupakan contoh cara mengimpor data gambar ke dalam array Numpy menggunakan library Matplotlib dan PIL. Dalam kode tersebut, gambar dengan format jpg diimpor dan disimpan dalam variabel `img` menggunakan fungsi `imread()`. Selanjutnya, gambar tersebut diubah menjadi array Numpy dengan mengambil nilai setiap piksel gambar pada channel warna merah (R) menggunakan slicing pada variabel `img`. Kemudian, array Numpy tersebut ditampilkan sebagai gambar dengan menggunakan fungsi `imshow()` dari Matplotlib. Dalam tampilan gambar tersebut, colormap "hot" digunakan untuk memberikan tampilan warna pada gambar.

Code #6: Memanipulasi eksposur gambar

```
#Memanipulasi eksposur gambar

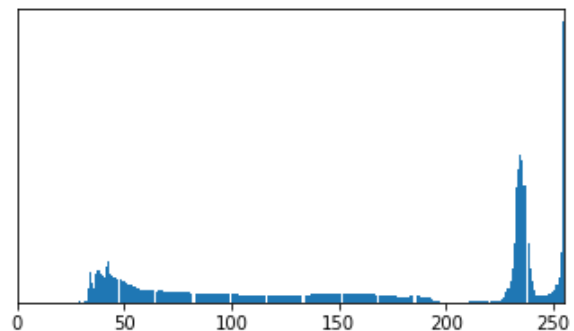
import numpy as np
import matplotlib.pyplot as plt
import skimage.exposure as skie

img = plt.imread("/content/drive/MyDrive/Foto/img (1).jpg")

def show(img) :
    #Display the image.
    fig, (ax1, ax2) = plt.subplots(1,2,figsize=(12,3))
    ax1.imshow(img, cmap=plt.cm.gray)
    ax1.set_axis_off()

    #Display the histogram.
    ax2.hist(img.ravel(),lw=0, bins=256)
    ax2.set_xlim(0, img.max())
    ax2.set_yticks([])
    plt.show()
    show(img)

show(skie.rescale_intensity(img, in_range=(0.4, .95), out_range=(0,1)))
```



Kode di atas menunjukkan bagaimana cara memanipulasi eksposur gambar menggunakan library skimage pada Python. Pertama, gambar diimpor menggunakan fungsi `plt.imread` dari direktori yang ditentukan. Kemudian, fungsi `show` digunakan untuk menampilkan gambar dan histogramnya dalam satu jendela. Setelah itu, fungsi `skie.rescale_intensity` digunakan untuk mengubah rentang nilai intensitas piksel pada gambar. Pada contoh di atas, nilai rentang masukan (`in_range`) yang digunakan adalah 0.4 dan 0.95, sedangkan rentang keluaran (`out_range`) adalah 0 dan 1. Dengan memanipulasi rentang nilai piksel pada gambar, hasilnya menjadi lebih jelas dan lebih mudah dilihat.