Institut Teknologi Nasional

VISUAL INNOVATIONS:

MODUL INOVATIF PEMBELAJARAN PEMROGRAMAN COMPUTER VISION

MODUL TEXTURE-BASED FEATURE DESCRIPTOR

2023

PROGRAM STUDI INFOMATIKA

INSTITUT TEKNOLOGI NASIONAL BANDUNG

By: Irma Amelia Dewi

DAFTAR ISI

KONFIGURASI GOOGLE COLABORATORY

Deskripsi

Google colab merupakan sebuah layanan dari google yang memungkinkan kita untuk menulis dan mengeksekusi Bahasa pemprograman Python pada browser dengan :

- Tidak dibutuhkan konfigurasi
- Akses gratis terhadap GPU
- Mudah di bagikan

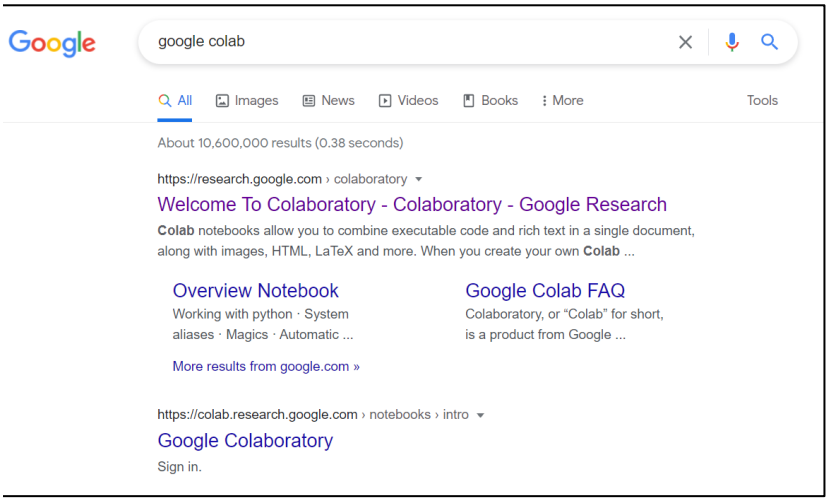
Google colab berjalan pada cloud service, oleh karena itu, komputer kita wajib terhubung ke jaringan internet.

Estimasi waktu 10 menit

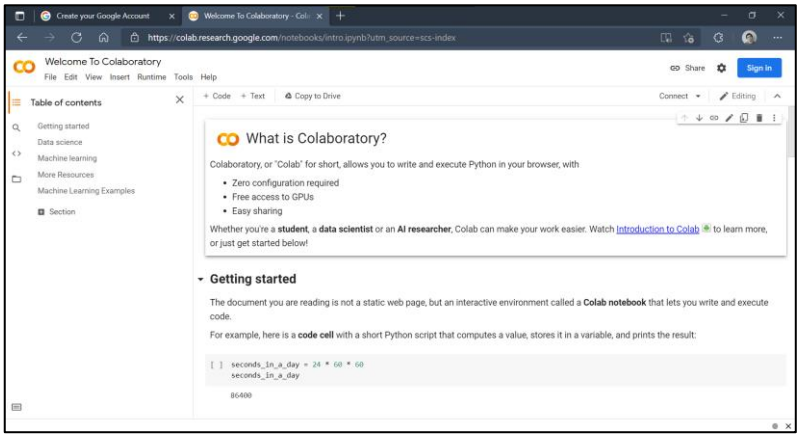
Prerequisite Siapkan akun google.

Alur Proses

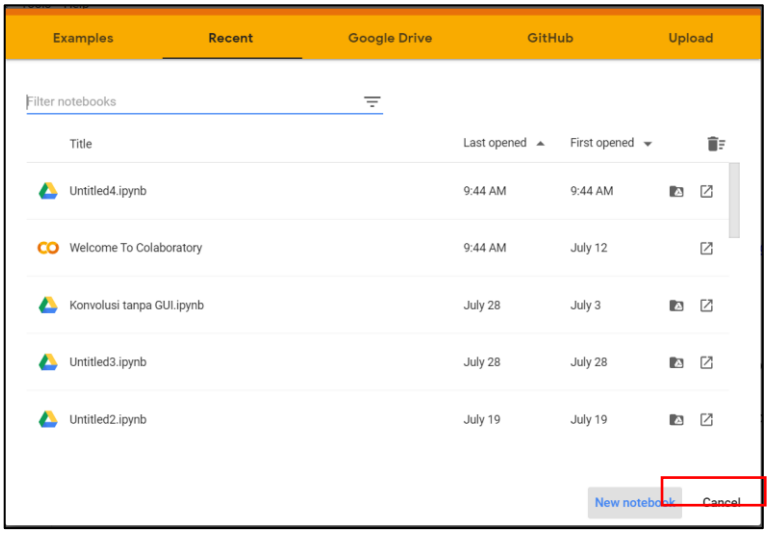
1. Buka browser, lakukan pencarian dengan kata kunci "google colab" kemudian pilih yang paling atas https://colab.research.google.com/



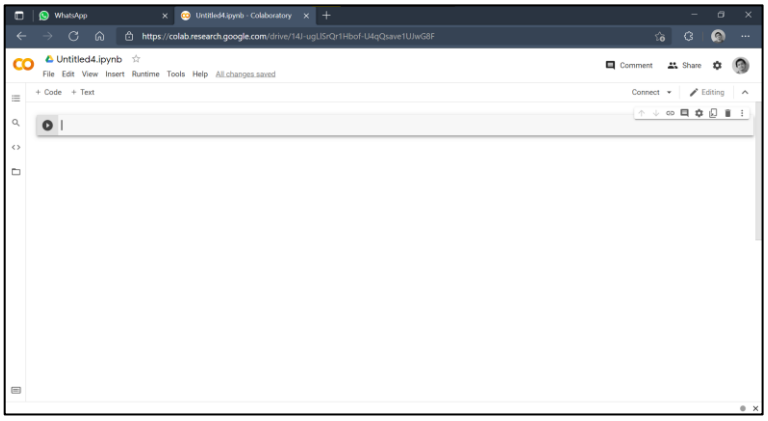
2. Jika belum sign in, sign in menggunakan akun google kalian



3. Untuk membuat dokumen baru, pilih "New notebook".



4. Selesai, google colab siap digunakan



E. TEXTURE-BASED FEATURE DESCRIPTORS

PRAKTEK E1- LOCAL BINARY PATTERN

Deskripsi	***Local Binary Pattern (LBP)*** adalah operator tekstur sederhana namun sangat efisien yang melabeli piksel gambar dengan membatasi lingkungan setiap piksel dan menganggap hasilnya sebagai bilangan biner.
Library	**OpenCV2** merupakan *library computer vision* yang dapat digunakan sebagai *library* dalam *digital image processing* **Numpy** merupakan library yang melakukan berbagai operasi matematika pada array
Estimasi waktu	10 menit
Prerequisite	1. Membuka notebook google colab 2. Menyediakan sebuah citra yang sudah di upload ke dalam penyimpanan sesi google colab
Alur Proses	1. Import library yang dibutuhkan 2. Baca file gambar yang akan di load ke sistem 3. Lakukan algoritma LBP pada gambar 4. Tampilkan gambar hasil dari LBP
Listing program	Pada halaman *colab* dapat diketikkan coding untuk menampilkan LBP

```
import cv2
 2. from google.colab.patches import cv2_imshow
 3. import numpy as np
 4.
 5. #Local Binary Pattern
 6. def get_pixel(img, center, x, y):
 7.
 8.   new_value = 0
 9.
10.   try:
11.     # If local neighbourhood pixel value is greater than or equal to center pixel values
then set it to 1
12.     if img[x][y] >= center:
13.       new_value = 1
14.
```

```
 except:
            # Exception is required when neighbourhood value of a center pixel value is null i.e. values present at boundaries.
            pass

    return new_value

# Function for calculating LBP
def lbp_calculated_pixel(img, x, y):

    center = img[x][y]

    val_ar = []

    # top_left
    val_ar.append(get_pixel(img, center, x-1, y-1))

    # top
    val_ar.append(get_pixel(img, center, x-1, y))

    # top_right
    val_ar.append(get_pixel(img, center, x-1, y + 1))

    # right
    val_ar.append(get_pixel(img, center, x, y + 1))

    # bottom_right
    val_ar.append(get_pixel(img, center, x + 1, y + 1))

    # bottom
    val_ar.append(get_pixel(img, center, x + 1, y))

    # bottom_left
    val_ar.append(get_pixel(img, center, x + 1, y-1))

    # left
    val_ar.append(get_pixel(img, center, x, y-1))

    # convert binary values to decimal
    power_val = [1, 2, 4, 8, 16, 32, 64, 128]

    val = 0

    for i in range(len(val_ar)):val += val_ar[i] * power_val[i]

    return val

height, width, _ = ori_img.shape
# Convert RGB image to gray image
img_gray = cv2.cvtColor(ori_img, cv2.COLOR_BGR2GRAY)

# Create a numpy array as the same height and width of RGB image
img_lbp = np.zeros((height, width),np.uint8)

for i in range(0, height):
    for j in range(0, width):
            img_lbp[i, j] = lbp_calculated_pixel(img_gray, i, j)

cv2_imshow(ori_img) #Original Image
cv2_imshow(img_lbp)

```

PRAKTEK E2- SIFT (SCALE INVARIANT FEATURE TRANSFORM)

Deskripsi	***Algoritma SIFT (Scale Invariant Feature Transform)*** adalah teknik Computer Vision yang digunakan untuk deteksi dan deskripsi fitur. Mendeteksi poin atau fitur kunci khusus dalam gambar yang tahan terhadap perubahan skala, rotasi, dan transformasi affine. SIFT bekerja dengan mengidentifikasi key points berdasarkan ekstrem intensitas lokalnya dan deskriptor komputasi yang menangkap informasi gambar lokal di sekitar titik kunci tersebut. Deskriptor ini kemudian dapat digunakan untuk tugas-tugas seperti pencocokan gambar, pengenalan objek, dan pengambilan gambar.
Library	**OpenCV2** merupakan *library computer vision* yang dapat digunakan sebagai *library* dalam *digital image processing* **Numpy** merupakan library yang melakukan berbagai operasi matematika pada array
Estimasi waktu	10 menit
Prerequisite	1. Membuka notebook google colab 3. Menyediakan sebuah citra yang sudah di upload ke dalam penyimpanan sesi google colab
Alur Proses	1. Import library yang dibutuhkan 5. Baca file gambar yang akan di load ke sistem 6. Lakukan algoritma SIFT 7. Tampilkan gambar hasil dari SIFT
Listing program	Pada halaman *colab* dapat diketikkan coding untuk menampilkan SIFT

```
#SIFT

# read the images
img1 = cv2.imread('book.jpg')
img2 = cv2.imread('table.jpg')
# convert images to grayscale
img1 = cv2.cvtColor(img1, cv2.COLOR_BGR2GRAY)
img2 = cv2.cvtColor(img2, cv2.COLOR_BGR2GRAY)
# create SIFT object
sift = cv2.xfeatures2d.SIFT_create()
# detect SIFT features in both images
keypoints_1, descriptors_1 = sift.detectAndCompute(img1,None)
keypoints_2, descriptors_2 = sift.detectAndCompute(img2,None)

# create feature matcher
bf = cv2.BFMatcher(cv2.NORM_L1, crossCheck=True)
# match descriptors of both images
matches = bf.match(descriptors_1,descriptors_2)

# sort matches by distance
matches = sorted(matches, key = lambda x:x.distance)
# draw first 50 matches
matched_img = cv2.drawMatches(img1, keypoints_1, img2, keypoints_2, matches[:50], img2,
flags=2)

# show the image
cv2_imshow(matched_img)
# save the image
cv2.imwrite("matched_images.jpg", matched_img)
cv2.waitKey(0)
cv2.destroyAllWindows()

```