

TUGAS 5
VISI KOMPUTER DAN PENGOLAHAN CITRA



Oleh :

Silfiana Nur Hamida
(1223800005)

Membahas tentang:
“Soal Ujian Akhir Semester”

PROGRAM PASCASARJANA TEKNIK INFORMATIKA DAN KOMPUTER
POLITEKNIK ELEKTRONIKA NEGERI SURABAYA

2023/2024

Soal UAS

1. Jelaskan tentang template matching menggunakan SAD dan SSD!
Buat script program sederhana untuk SAD dan SSD, dikumpulkan dan demo minggu depan!

Template matching menggunakan SAD (Sum Absolute Difference) merupakan sebuah teknik yang digunakan dalam pengolahan citra untuk menemukan bagian bagian kecil dari gambar yang cocok dengan gambar sebenarnya menggunakan proses filtering

$$h[m,n] = \sum_{k,l} |g[k,l] - f[m+k,n+l]|$$

Template matching menggunakan SSD (Sum Square Differe) merupakan sebuah teknik yang digunakan dalam pengolahan citra untuk enemukan bagian bagian kecil dari gambar yang cocok dengan gambar aslinya yang menggunakan proses filtering.

$$h[m,n] = \sum_{k,l} (g[k,l] - f[m+k,n+l])^2$$

- a. Source Code SAD

```
import cv2

def calculate_SAD(image1, image2):
    img1 = cv2.imread(image1, cv2.IMREAD_GRAYSCALE)
    img2 = cv2.imread(image2, cv2.IMREAD_GRAYSCALE)

    if img1 is None or img2 is None:
        print("Gambar tidak dapat dibaca. Pastikan path file benar.")
        return
    if img1.shape != img2.shape:
        print("Ukuran gambar tidak sama.")
        return

    # Hitung SAD antara kedua gambar
    sad_value = cv2.absdiff(img1, img2).sum()

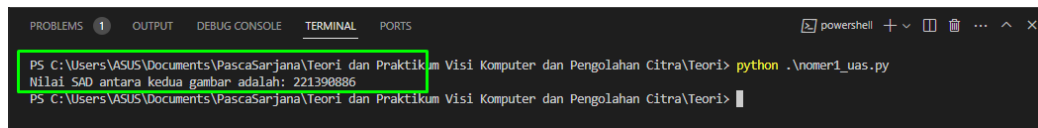
    return sad_value

if __name__ == "__main__":
    image_path1 = 'mybest.jpg'
    image_path2 = 'myfriend.jpg'

    # Hitung SAD antara kedua gambar
    sad = calculate_SAD(image_path1, image_path2)

    if sad is not None:
        print(f"Nilai SAD antara kedua gambar adalah: {sad}")
```

b. Output Program



```
PS C:\Users\ASUS\Documents\PascaSarjana\Teori dan Praktikum Visi Komputer dan Pengolahan Citra\Teori> python .\nomer1_uas.py
Nilai SAD antara kedua gambar adalah: 221390886
PS C:\Users\ASUS\Documents\PascaSarjana\Teori dan Praktikum Visi Komputer dan Pengolahan Citra\Teori>
```

c. Source Code SSD

```
import cv2
import numpy as np

def ssd(image1, image2):
    if image1.shape != image2.shape:
        raise ValueError("Ukuran citra tidak sama")

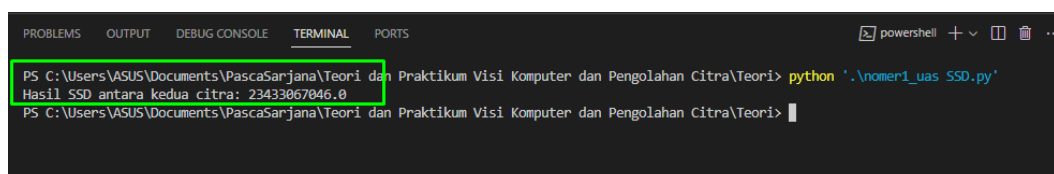
    squared_diff = np.square(image1.astype(np.float64) -
image2.astype(np.float64))
    ssd_value = np.sum(squared_diff)

    return ssd_value

# Baca dua citra
img1 = cv2.imread('mybest.jpg' , cv2.IMREAD_GRAYSCALE)
img2 = cv2.imread('myfriend.jpg' , cv2.IMREAD_GRAYSCALE)

# Hitung SSD antara kedua citra
ssd_value = ssd(img1, img2)
print(f"Hasil SSD antara kedua citra: {ssd_value}")
```

d. Output Program



```
PS C:\Users\ASUS\Documents\PascaSarjana\Teori dan Praktikum Visi Komputer dan Pengolahan Citra\Teori> python .\nomer1_uas SSD.py
Hasil SSD antara kedua citra: 23433067046.0
PS C:\Users\ASUS\Documents\PascaSarjana\Teori dan Praktikum Visi Komputer dan Pengolahan Citra\Teori>
```

2. Jelaskan tentang filter bank dan image pyramid!

Buat script program sederhana untuk filter diatas, dikumpulkan dan demo minggu depan!

Filter bank merupakan kumpulan template atau kernel filter yang dapat digunakan untuk proses template matching atau ekstraksi fitur.

$$F(r, \sigma, \tau) = F_0(\sigma, \tau) + \cos\left(\frac{\pi r r}{\sigma}\right) e^{-\frac{r^2}{2\sigma^2}}$$

Image pyramid merupakan salah satu teknik dalam pengolahan citra yang digunakan untuk merepresentasikan citra dalam skala yang berbeda.

a. Source code

```
import cv2
import numpy as np

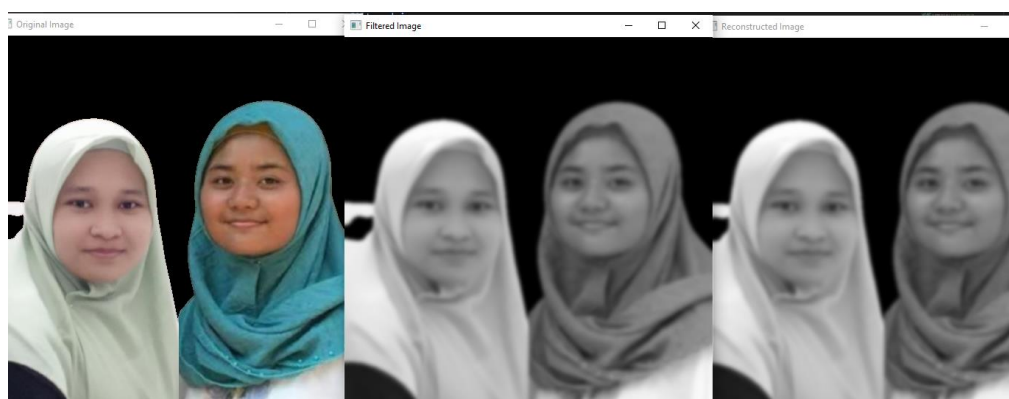
def apply_gaussian_filter(image):
    if len(image.shape) > 2:
        image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    # Filter Gaussian
    filtered_image = cv2.GaussianBlur(image, (15, 15), 0)
    return filtered_image

# Fungsi filter
def reconstruct_image(filtered_image, original_shape):
    reconstructed_image = cv2.resize(filtered_image,
    original_shape[:2][::-1])

    return reconstructed_image
input_image = cv2.imread('bestie.png')
original_shape = input_image.shape

filtered_image = apply_gaussian_filter(input_image)
reconstructed_image = reconstruct_image(filtered_image,
original_shape)
cv2.imshow('Original Image', input_image)
cv2.imshow('Filtered Image', filtered_image)
cv2.imshow('Reconstructed Image', reconstructed_image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

b. Output Program



3. Jelaskan tentang metode ohlander!

Buat script program sederhana untuk metode diatas, dikumpulkan dan demo minggu depan!

Metode ohlander merupakan suatu teknik segmentasi citra yang digunakan untuk mengelompokkan piksel dalam citra berdasarkan kriteria tertentu

a. Source code

```
import cv2
import numpy as np

# Fungsi untuk melakukan segmentasi menggunakan metode Ohlander's
Recursive Histogram-Based Clustering
def ohlander_clustering(image, threshold):
    if len(image.shape) > 2:
        gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    else:
        gray_image = image.copy()

    # Mendapatkan histogram dari gambar
    hist = cv2.calcHist([gray_image], [0], None, [256], [0, 256])

    # Mencari nilai untuk clustering
    split_value = 0
    max_val = np.max(hist)
    for i in range(255, 0, -1):
        if hist[i] > threshold * max_val:
            split_value = i
            break

    # Segmentasi gambar
    segmented_image = np.zeros_like(gray_image)
    segmented_image[gray_image >= split_value] = 255

    return segmented_image
input_image = cv2.imread('dot.jpg')

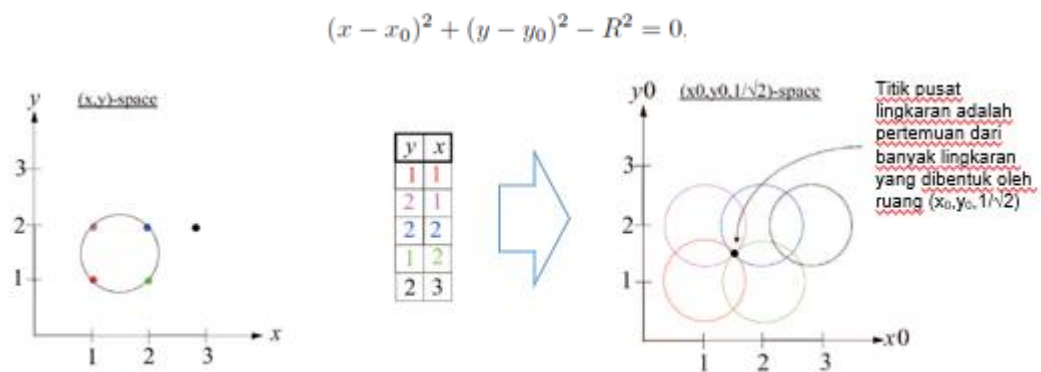
threshold_value = 0.7
segmented_image = ohlander_clustering(input_image, threshold_value)
cv2.imshow('Original Image', input_image)
cv2.imshow('Segmented Image', segmented_image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

b. Output Program



4. Jelaskan cara perhitungan transformasi hough untuk mendeteksi lingkaran
Buat script program sederhana untuk metode diatas, dikumpulkan dan demo minggu depan!

Transformasi hough merupakan suatu teknik yang digunakan untuk mendeteksi garis dan lingkaran dalam suatu citra menggunakan persamaan lingkaran dengan titik pusat (x_0, y_0) dan jari jari (R) pada titik ruang (x,y) .



a. Source Code

```
import cv2
import numpy as np

# Baca gambar
image = cv2.imread('bestie.png')
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
blur = cv2.GaussianBlur(gray, (5, 5), 0) # Blur mengurangi noise
edges = cv2.Canny(blur, 50, 150) # Deteksi tepi dengan Canny edge detection

# Transformasi Hough
circles = cv2.HoughCircles(edges, cv2.HOUGH_GRADIENT, dp=1,
minDist=50,
                                param1=100, param2=30, minRadius=20,
maxRadius=100)

if circles is not None:
```

```
circles = np.uint16(np.around(circles))

for circle in circles[0, :]:
    center = (circle[0], circle[1])
    radius = circle[2]
    cv2.circle(image, center, radius, (0, 255, 0), 2)
cv2.imshow('Hasil Deteksi Lingkaran dengan Transformasi Hough',
image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

b. Output Program

