LAPORAN TEORI

PENGOLAHAN CITRA DIGITAL

NAMA : Fazli Haqqi M Ramadhani

NIM : 202331233

KELAS : C

DOSEN : Ir. Darma Rusjdi, M.Kom

NO.PC : 12

ASISTEN : 1. Abdur Rasyid Ridho

2. Rizqy Amanda

3. Kashrina Masyid Azka

4. Izzat Islami Kagapi

INSTITUT TEKNOLOGI PLN

TEKNIK INFORMATIKA

2025

1. **Apa itu operasi konvolusi dalam konteks pengolahan citra digital?**
   Operasi konvolusi adalah proses matematika yang digunakan untuk memanipulasi nilai-nilai piksel pada suatu citra digital dengan cara mengalikan piksel-piksel sekitarnya menggunakan sebuah kernel atau filter. Hasil perkalian tersebut dijumlahkan dan digunakan sebagai nilai baru dari piksel yang sedang diproses. Konvolusi banyak digunakan untuk efek seperti blur, edge detection, sharpening, dan lain-lain.
2. **Perbedaan antara filter rata-rata (mean filter) dan filter median (median filter)**

- Mean Filter: rata-rata nilai piksel, cocok untuk blur umum.
- Median Filter: nilai tengah piksel, efektif hilangkan noise (salt & pepper).
- Median lebih baik mempertahankan tepi citra.

3. **Langkah-langkah proses konvolusi pada satu piksel**

- Pilih kernel/filter, misalnya 3x3 matriks.
- Letakkan kernel di atas piksel target (pusat kernel berada di piksel yang ingin diproses).
- Kalikan nilai-nilai dalam kernel dengan nilai piksel yang sesuai di bawahnya.
- Jumlahkan hasil perkalian tersebut.
- Simpan hasilnya sebagai nilai baru untuk piksel target (bisa langsung, atau setelah normalisasi).

4. **Mengapa konvolusi penting dalam pengolahan citra & CNN (Deep Learning)?**

- Konvolusi adalah dasar dari CNN (Convolutional Neural Networks).
- CNN memanfaatkan filter untuk menangkap pola penting seperti garis, tepi, tekstur, objek dari gambar.
- Dibandingkan metode klasik, CNN belajar sendiri filter optimal dari data, sehingga hasil lebih akurat.
- Signifikansinya:
  - Efisiensi parameter (dibanding fully connected layer).
  - Invarian translasi: mengenali objek meskipun posisi sedikit berubah.
  - Hierarki fitur: dari fitur sederhana (edge) → kompleks (wajah, objek).

5. **Aplikasi dan contoh nyata penggunaan konvolusi**

- Penghapusan noise: Gunakan median filter untuk memperbaiki gambar dari CCTV buram.
- Deteksi tepi (edge detection): Digunakan dalam mesin fotokopi, pemrosesan X-ray, kamera HP.
- Pengenalan wajah & biometrik: CNN berbasis konvolusi mendeteksi wajah untuk sistem keamanan.
- Pendeteksian objek pada kendaraan otonom.
- Pengolahan gambar medis: deteksi tumor, pembuluh darah, dll.

LAPORAN PRAKTIKUM

PENGOLAHAN CITRA DIGITAL

NAMA : Fazli Haqqi M Ramadhani

NIM : 202331233

KELAS : C

DOSEN : Ir. Darma Rusjdi, M.Kom

NO.PC : 12

ASISTEN : 1. Abdur Rasyid Ridho

2. Rizqy Amanda

3. Kashrina Masyid Azka

4. Izzat Islami Kagapi

INSTITUT TEKNOLOGI PLN

TEKNIK INFORMATIKA

2025

1. **Import Library**

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

# 202331233_Fazli Haqqi M Ramadhani
```

Penjelasan:
Saya mulai dengan import library yang dibutuhkan:

- cv2 untuk pengolahan citra (menggunakan OpenCV),
- numpy untuk operasi matematika, terutama array,
- matplotlib.pyplot untuk menampilkan gambar.

2. **Membaca, Menyiapkan Gambar, dan Konversi Warna**

```
# === Membaca gambar dan mengonversi ke format RGB dan grayscale
gambar_mobil = cv2.imread('desktop-wallpaper-mobil-sport-keren-mobil-sport-full.jpg')
if gambar_mobil is None:
    raise FileNotFoundError("Gambar mobil tidak ditemukan. Pastikan path-nya benar.")

mobil_rgb = cv2.cvtColor(gambar_mobil, cv2.COLOR_BGR2RGB)
mobil_gray = cv2.cvtColor(gambar_mobil, cv2.COLOR_BGR2GRAY)

# 202331233_Fazli Haqqi M Ramadhani
```

Penjelasan:
Kode ini membaca gambar dari file JPG. Saya tambahkan pengecekan if gambar_mobil is None: agar kalau file tidak ditemukan, akan muncul error dengan pesan jelas.

- mobil_rgb: digunakan untuk ditampilkan di matplotlib (karena matplotlib pakai RGB, sementara OpenCV default-nya BGR).
- mobil_gray: konversi ke grayscale, diperlukan karena deteksi tepi biasanya lebih baik dilakukan pada citra hitam-putih.

3. **Filter Rata-rata**

```
# === Filter Rata-rata (Blur)
blur_rata = cv2.blur(mobil_rgb, (7, 7))

# 202331233_Fazli Haqqi M Ramadhani
```

Penjelasan:
Saya menggunakan cv2.blur() dengan kernel 7x7. Fungsi ini menghitung rata-rata dari setiap blok 7x7 piksel dan menggantinya ke piksel pusat. Hasilnya adalah gambar yang tampak lebih halus dan sedikit buram.

4. **Filter Median**

```
# === Filter Median
blur_median = cv2.medianBlur(mobil_rgb, 7)

# 202331233_Fazli Haqqi M Ramadhani
```

Penjelasan:
Berbeda dengan blur rata-rata, medianBlur() mengambil nilai median dari blok 7x7. Median ini lebih bagus untuk menghilangkan noise salt and pepper, karena tidak mengubah tepi gambar sebanyak average blur.

5. Deteksi Tepi

```
# === Deteksi Tepi (Edge Detection)
kernel_edge = np.array([[-1, -1, -1],
                        [-1,  8, -1],
                        [-1, -1, -1]])
mobil_tepi = cv2.filter2D(mobil_gray, -1, kernel_edge)

# 202331233_Fazli Haqqi M Ramadhani
```

Penjelasan:
Di bagian ini saya membuat kernel konvolusi 3x3 untuk mendeteksi tepi. Nilai 8 di tengah membuat perbedaan piksel lebih terlihat. Semua -1 di sekelilingnya membuat perubahan kontras muncul.
Lalu saya terapkan ke gambar grayscale pakai cv2.filter2D().

6. Menampilkan Semua Gambar

```
# === Menampilkan hasil dalam bentuk subplot 2x2
fig, axes = plt.subplots(2, 2, figsize=(12, 10))

axes[0, 0].imshow(mobil_rgb)
axes[0, 0].set_title("1. Gambar Asli (Mobil Sport)")
axes[0, 0].axis('off')

axes[0, 1].imshow(blur_rata)
axes[0, 1].set_title("2. Filter Rata-rata (Blur Lembut)")
axes[0, 1].axis('off')

axes[1, 0].imshow(blur_median)
axes[1, 0].set_title("3. Filter Median (Penghilang Noise)")
axes[1, 0].axis('off')

axes[1, 1].imshow(mobil_tepi, cmap='gray')
axes[1, 1].set_title("4. Filter Tepi (Deteksi Batas Objek)")
axes[1, 1].axis('off')

plt.tight_layout()
plt.show()

# 202331233_Fazli Haqqi M Ramadhani
```

Penjelasan:
Bagian ini membuat kanvas berisi 4 kolom gambar (2 baris x 2 kolom), dengan ukuran yang cukup besar supaya semua hasil terlihat jelas.
Kemudian, keempat subplot diisi dengan:

- [0,0]: Gambar mobil asli (RGB),
- [0,1]: Hasil filter blur rata-rata,
- [1,0]: Hasil median filter,
- [1,1]: Hasil deteksi tepi menggunakan kernel konvolusi.

Setiap gambar diberi judul masing-masing agar penonton tahu filter apa yang diterapkan, dan axis('off') digunakan supaya sumbu X dan Y disembunyikan demi tampilan yang bersih. Akhirnya, plt.tight_layout() dipakai untuk merapikan tata letak antar gambar supaya tidak saling bertumpuk, lalu plt.show() digunakan untuk menampilkan keseluruhan hasil dalam satu jendela grafik.

Output Gambar

```
# 202331233_Fazli Haqqi M Ramadhani
```

7. Baca dan konversi gambar ke RGB

```
# Membaca dan mengubah gambar boneka ke format RGB

img_boneka = cv2.imread('boneka.jpg')
img_boneka_rgb = cv2.cvtColor(img_boneka, cv2.COLOR_BGR2RGB)

# 202331233_Fazli Haqqi M Ramadhani
```

Di bagian ini, saya mulai dengan membaca gambar boneka dari file. Karena cv2.imread() itu default-nya BGR, saya ubah dulu ke RGB supaya warnanya benar saat ditampilin di matplotlib.

8. Siapkan kernel sharpening

```
# Membuat kernel konvolusi (sharpening)
kernel_sharpen = np.array([[0, -1, 0],
                           [-1, 5, -1],
                           [0, -1, 0]])

# 202331233_Fazli Haqqi M Ramadhani
```

Di sini saya buat kernel konvolusi 3x3 untuk efek penajaman (sharpen). Nilai 5 di tengahnya bikin pixel pusat jadi lebih dominan, sedangkan -1 di sekelilingnya untuk menonjolkan kontras tepi objek.

9. Konversi ke grayscale

```
# Konversi ke grayscale sebelum dikonvolusi
boneka_gray = cv2.cvtColor(img_boneka, cv2.COLOR_BGR2GRAY)

# 202331233_Fazli Haqqi M Ramadhani
```

Sebelum dikasih filter, saya ubah dulu gambar ke **grayscale** karena proses konvolusi lebih optimal di citra hitam putih.

10. Proses konvolusi filter 2D

```
# Terapkan filter konvolusi sharpen
boneka_sharpened = cv2.filter2D(boneka_gray, -1, kernel_sharpen)

# 202331233_Fazli Haqqi M Ramadhani
```

Nah, di sini saya apply **filter sharpen** ke gambar grayscale-nya. Saya pakai fungsi filter2D dari OpenCV. Hasilnya nanti akan nunjukin garis tepi jadi lebih jelas dan tajam.

11. Tampilkan hasil dalam dua panel

```
# Menyiapkan subplot untuk menampilkan gambar asli dan hasil
fig, axs = plt.subplots(1, 2, figsize=(12, 6))

axs[0].imshow(img_boneka_rgb)
axs[0].set_title('1. Citra Asli (Boneka Sapi)')
axs[0].axis('off')

axs[1].imshow(boneka_sharpened, cmap='gray')
axs[1].set_title('2. Hasil Konvolusi Filter 2D (Sharpen)')
axs[1].axis('off')

plt.tight_layout()
plt.show()

# 202331233_Fazli Haqqi M Ramadhani
```

Terakhir, saya pakai matplotlib buat **nampilin dua gambar sekaligus**:

- Kiri: Gambar asli warna
- Kanan: Gambar hasil sharpen hitam putih

Saya tambahkan tight_layout() supaya tampilannya rapi dan gak saling numpuk.

Output Gambar