LAPORAN TEORI

PENGOLAHAN CITRA DIGITAL

NAMA : Kevin Naufal Raditya

NIM : 202331236

KELAS : C

DOSEN : Ir. Darma Rusjdi, M.Kom

NO.PC :

ASISTEN : 1. Abdur Rasyid Ridho

2. Rizqy Amanda

3. Kashrina Masyid Azka

4. Izzat Islami Kagapi

INSTITUT TEKNOLOGI PLN

SISTEM INFORMASI

2025

1. Apa itu Operasi Konvolusi?
   Konvolusi adalah operasi matematis fundamental dalam pengolahan citra yang menggunakan sebuah matriks kecil bernama **kernel** (filter) untuk memodifikasi sebuah citra. Kernel ini digeser ke seluruh area citra untuk melakukan berbagai tugas seperti penghalusan (blur), penajaman (sharpen), dan deteksi fitur.

2. Perbedaan Filter Rata-rata vs. Median\
   - Filter Rata-rata (Mean): Menghitung nilai rata-rata dari piksel di bawah kernel. Efektif untuk mengurangi noise umum (Gaussian noise) tetapi mengaburkan tepi objek.
   - Filter Median: Mengambil nilai tengah dari piksel di bawah kernel setelah diurutkan. Sangat efektif untuk menghilangkan noise salt-and-pepper dan mampu mempertahankan ketajaman tepi gambar.

3. Langkah-langkah Konvolusi pada Satu Piksel
   - Tempatkan kernel di atas piksel target pada citra input.
   - Kalikan setiap elemen kernel dengan nilai piksel yang bersesuaian di bawahnya.
   - Jumlahkan semua hasil perkalian tersebut.
   - Tetapkan hasil penjumlahan sebagai nilai piksel baru pada citra output.

4. Mengapa Konvolusi Sangat Fundamental?
   - Dalam Pengolahan Citra: Menjadi dasar untuk berbagai filter (deteksi tepi, blur, dll.).
   - Dalam Deep Learning (CNN): Menjadi inti dari Convolutional Neural Networks. Alih-alih dirancang manual, CNN mempelajari filter (kernel) terbaik secara otomatis untuk mengenali pola kompleks, mulai dari garis sederhana hingga objek utuh seperti wajah atau mobil.

5. Aplikasi Utama Konvolusi
   - Fotografi: Mode potret pada smartphone menggunakan blur (konvolusi) untuk mengaburkan latar belakang.
   - Keamanan: Sistem pengenalan wajah (facial recognition) menggunakan CNN untuk mendeteksi fitur wajah.
   - Visi Komputer: Mobil otonom menggunakan deteksi tepi untuk mengenali marka jalan.
   - Pencitraan Medis: Meningkatkan kualitas gambar MRI atau CT scan untuk membantu diagnosis.

LAPORAN PRAKTIKUM

PENGOLAHAN CITRA DIGITAL

NAMA : Kevin Naufal Raditya

NIM : 202331236

KELAS : C

DOSEN : Ir. Darma Rusjdi, M.Kom

NO.PC :

ASISTEN : 1. Abdur Rasyid Ridho

2. Rizqy Amanda

3. Kashrina Masyid Azka

4. Izzat Islami Kagapi

INSTITUT TEKNOLOGI PLN

SISTEM INFORMASI

2025

1. Penjelasan Sel 1

```
[2]  # Import library yang diperlukan
     import cv2
     import numpy as np
     import matplotlib.pyplot as plt

     # Fungsi bantuan untuk menampilkan gambar di Colab
     # Karena cv2.imshow() tidak berfungsi di Colab, kita pakai Matplotlib
     def tampilkan_gambar(judul, gambar):
       """Menampilkan sebuah gambar dengan judul menggunakan Matplotlib."""
       # Konversi gambar dari BGR (format OpenCV) ke RGB (format Matplotlib)
       gambar_rgb = cv2.cvtColor(gambar, cv2.COLOR_BGR2RGB)
       plt.imshow(gambar_rgb)
       plt.title(judul)
       plt.axis('off') # Menyembunyikan sumbu x dan y

     def tampilkan_gambar_gray(judul, gambar):
       """Menampilkan gambar grayscale dengan judul."""
       plt.imshow(gambar, cmap='gray')
       plt.title(judul)
       plt.axis('off')
```

Kevin Naufal Raditya
202331236

Tujuan Utama Sel Ini: Sel ini bertujuan untuk **mempersiapkan lingkungan kerja** Anda. Sebelum kita bisa mengolah gambar, kita perlu memuat semua "alat" (library) yang dibutuhkan dan membuat fungsi-fungsi pembantu agar kode kita lebih rapi dan efisien.

Penjelasan Baris per Baris:

- import cv2: Baris ini mengimpor library OpenCV (sering disebut cv2). Library ini adalah inti dari pengolahan citra di Python, menyediakan fungsi untuk membaca gambar, menerapkan filter, mengubah warna, dan banyak lagi.
- import numpy as np: Mengimpor library NumPy dan memberinya alias np. NumPy sangat penting karena gambar pada dasarnya direpresentasikan sebagai matriks (array) angka. NumPy memungkinkan kita membuat dan memanipulasi matriks ini dengan mudah, seperti saat kita membuat kernel konvolusi.
- import matplotlib.pyplot as plt: Mengimpor library Matplotlib, khususnya modul pyplot dengan alias plt. Di Google Colab, kita tidak bisa menggunakan cv2.imshow() untuk menampilkan gambar. Matplotlib adalah standar untuk membuat plot dan menampilkan gambar langsung di dalam notebook.
- def tampilkan_gambar(...): Kita mendefinisikan sebuah fungsi pembantu bernama tampilkan_gambar.
    - Tujuannya: Agar kita tidak perlu menulis kode yang sama berulang-ulang setiap kali ingin menampilkan gambar berwarna.
    - cv2.cvtColor(gambar, cv2.COLOR_BGR2RGB): Ini adalah baris yang sangat penting. OpenCV membaca gambar dalam format warna BGR (Blue, Green, Red), sedangkan Matplotlib menampilkannya dalam format RGB (Red, Green, Blue). Baris ini mengonversi format warna agar gambar yang ditampilkan tidak aneh (misalnya, area biru menjadi oranye).

- plt.imshow(...), plt.title(...), plt.axis('off'): Baris-baris ini menggunakan Matplotlib untuk menampilkan gambar, memberinya judul, dan menghilangkan sumbu X-Y yang tidak perlu.

- **def tampilkan_gambar_gray(...)**: Fungsi ini mirip dengan yang sebelumnya, tetapi khusus untuk menampilkan gambar **grayscale** (hitam-putih). cmap='gray' memberitahu Matplotlib untuk menggunakan palet warna abu-abu.

Hasil Akhir: Setelah menjalankan sel ini, tidak ada output visual yang muncul. Namun, semua library dan fungsi pembantu sudah dimuat ke dalam memori dan siap digunakan di sel-sel berikutnya.

2. Penjelasan Sel 2

```
[3] # Ganti nama file ini jika nama file Anda berbeda
    NAMA_FILE_MAKANAN = 'kerupuk.jpg'

    # 1. Memuat gambar
    img_makanan = cv2.imread(NAMA_FILE_MAKANAN)

    if img_makanan is None:
      print(f"Error: Gambar '{NAMA_FILE_MAKANAN}' tidak ditemukan. Pastikan sudah diunggah.")
    else:
      # 2. Aplikasi Filter Rata-rata (Mean Filter)
      # Menggunakan fungsi cv2.blur() yang lebih efisien dengan kernel 7x7
      makanan_rata_rata = cv2.blur(img_makanan, (7, 7))

      # 3. Aplikasi Filter Median
      # Ukuran kernel harus ganjil, misal 7
      makanan_median = cv2.medianBlur(img_makanan, 7)

      # 4. Aplikasi Filter Batas (Edge Detection) menggunakan Laplacian
      # Filter batas lebih efektif pada gambar grayscale
      img_makanan_gray = cv2.cvtColor(img_makanan, cv2.COLOR_BGR2GRAY)
      laplacian = cv2.Laplacian(img_makanan_gray, cv2.CV_64F)
      makanan_batas = cv2.convertScaleAbs(laplacian)

      # 5. Menampilkan semua hasil dalam satu plot
      plt.figure(figsize=(12, 10))

      plt.subplot(2, 2, 1)
      tampilkan_gambar("Gambar Asli", img_makanan)

      plt.subplot(2, 2, 2)
      tampilkan_gambar("Filter Rata-rata", makanan_rata_rata)

      plt.subplot(2, 2, 3)
      tampilkan_gambar("Filter Median", makanan_median)

      plt.subplot(2, 2, 4)
      tampilkan_gambar_gray("Filter Batas (Laplacian)", makanan_batas)

      plt.suptitle("Hasil Konvolusi pada Gambar Makanan", fontsize=16)
      plt.show()
```

Kevin Naufal Raditya
202331236

Tujuan Utama Sel Ini: Sel ini adalah eksekusi dari **tugas pertama**: memuat gambar makanan, menerapkan tiga jenis filter konvolusi (rata-rata, median, dan batas), lalu menampilkan hasilnya secara berdampingan untuk perbandingan.

Penjelasan Baris per Baris:

- NAMA_FILE_MAKANAN = 'kerupuk.jpg': Mendefinisikan nama file dalam sebuah variabel agar mudah diganti jika diperlukan.
- img_makanan = cv2.imread(...): Membaca file gambar dari penyimpanan Colab dan memuatnya sebagai objek NumPy.
- if img_makanan is None:: Sebuah blok pengecekan eror yang baik. Jika gambar gagal dimuat (misalnya, salah nama atau belum diunggah), program akan mencetak pesan eror alih-alih langsung crash.

- makanan_rata_rata = cv2.blur(img_makanan, (7, 7)): Menerapkan filter rata-rata (mean filter). Fungsi cv2.blur() adalah cara cepat untuk melakukan konvolusi dengan kernel rata-rata. (7, 7) adalah ukuran kernel; semakin besar angkanya, semakin kabur (blur) hasilnya.
- makanan_median = cv2.medianBlur(img_makanan, 7): Menerapkan filter median. Filter ini sangat baik untuk menghilangkan noise "bintik-bintik" (salt-and-pepper). Angka 7 adalah ukuran kernel (harus ganjil).
- img_makanan_gray = cv2.cvtColor(...): Mengonversi gambar asli ke grayscale. Ini adalah langkah persiapan untuk filter batas, karena deteksi tepi biasanya bekerja pada perubahan intensitas (tingkat keabuan), bukan pada warna.
- laplacian = cv2.Laplacian(...): Menerapkan filter batas menggunakan operator Laplacian. Filter ini akan menyorot area di mana ada perubahan warna atau intensitas yang drastis, yaitu tepi objek. cv2.CV_64F adalah tipe data teknis untuk mencegah kehilangan informasi.
- makanan_batas = cv2.convertScaleAbs(...): Mengonversi hasil Laplacian kembali ke format gambar 8-bit yang dapat ditampilkan.
- plt.figure(...), plt.subplot(...), plt.show(): Blok ini bertanggung jawab untuk tata letak visual.
    - plt.figure(figsize=(12, 10)): Membuat "kanvas" besar untuk menampung semua gambar.
    - plt.subplot(2, 2, 1): Membuat grid 2x2 dan memilih posisi pertama (kiri atas) untuk gambar berikutnya.
    - Setelah itu, kita memanggil fungsi tampilkan_gambar dan tampilkan_gambar_gray yang kita buat di Sel 1 untuk menempatkan setiap gambar di posisinya masing-masing.
    - plt.show(): Menampilkan keseluruhan plot yang sudah kita susun.

Hasil Akhir: Output visual berupa sebuah grid 2x2 yang menampilkan: gambar asli, gambar hasil filter rata-rata (lebih kabur), gambar hasil filter median (noise berkurang, tepi terjaga), dan gambar hasil filter batas (hanya menampilkan garis-garis tepi objek).

3. Penjelasan Sel 3

```
# Ganti nama file ini jika nama file Anda berbeda
NAMA_FILE_BONEKA = 'boneka.jpg'

# 1. Memuat gambar boneka
img_boneka = cv2.imread(NAMA_FILE_BONEKA)

if img_boneka is None:
  print(f"Error: Gambar '{NAMA_FILE_BONEKA}' tidak ditemukan. Pastikan sudah diunggah.")
else:
  # 2. Membuat kernel kustom untuk filter 2D (contoh: kernel penajam/sharpening)
  kernel_penajam = np.array([
      [ 0, -1,  0],
      [-1,  5, -1],
      [ 0, -1,  0]
  ])

  # 3. Menerapkan konvolusi menggunakan cv2.filter2D()
  # ddepth = -1 berarti gambar output akan memiliki kedalaman (tipe data) yang sama dengan input
  boneka_tajam = cv2.filter2D(src=img_boneka, ddepth=-1, kernel=kernel_penajam)

  # 4. Menampilkan hasil
  plt.figure(figsize=(10, 5))

  plt.subplot(1, 2, 1)
  tampilkan_gambar("Boneka Asli", img_boneka)

  plt.subplot(1, 2, 2)
  tampilkan_gambar("Hasil Filter 2D (Penajam)", boneka_tajam)

  plt.suptitle("Konvolusi Filter 2D pada Gambar Boneka", fontsize=16)
  plt.show()
```

Kevin Naufal Raditya
202331236

Tujuan Utama Sel Ini: Sel ini adalah eksekusi dari **tugas kedua**: mendemonstrasikan penggunaan cv2.filter2D() dengan cara membuat **kernel konvolusi kustom** sendiri dan menerapkannya pada gambar boneka.

Penjelasan Baris per Baris:

- img_boneka = cv2.imread(...): Memuat gambar boneka sapi.
- kernel_penajam = np.array([...]): Ini adalah bagian inti dari sel ini. Kita membuat matriks 3x3 menggunakan NumPy. Matriks ini adalah kernel penajam (sharpening kernel).
    - Cara kerjanya: Nilai 5 di tengah akan memperkuat nilai piksel aslinya, sementara nilai -1 di sekitarnya akan mengurangi nilai piksel tetangganya. Efeknya, perbedaan antara piksel dan tetangganya menjadi lebih besar, membuat gambar terlihat lebih tajam.
- boneka_tajam = cv2.filter2D(...): Ini adalah fungsi konvolusi yang paling umum
    - src=img_sapi: Gambar input.
    - ddepth=-1: Parameter teknis yang berarti tipe data gambar output sama dengan input.
    - kernel=kernel_penajam: Di sinilah kita memberikan kernel kustom yang baru saja kita buat. cv2.filter2D akan menggeser kernel ini ke seluruh gambar dan melakukan perhitungan konvolusi.
- plt.figure(...), plt.subplot(...), tampilkan_gambar(...), plt.show(): Sama seperti sebelumnya, bagian ini mengatur tata letak untuk menampilkan gambar asli dan gambar yang sudah dipertajam secara berdampingan agar perbedaannya terlihat jelas.

Hasil Akhir: Output visual berupa dua gambar berdampingan: gambar boneka asli dan gambar boneka yang terlihat lebih detail dan tajam setelah diaplikasikan filter kustom menggunakan cv2.filter2D.