

LAPORAN AKHIR
DIGITAL IMAGE PROCESSING METODE MEDIAN FILTERING DAN
MORFOLOGI OPENING DALAM REDUKSI NOISE CITRA



Disusun Guna Memenuhi Tugas Akhir Mata Kuliah
Pengolahan Citra Digital

DISUSUN OLEH:

Kelompok 16:

Imro'atus Sholihah 23051204349

Najwa Vaida Isnani 23051204355

DOSEN PENGAMPU:

Dr. Ir. Ricky Eka Putra, S.Kom., M.Kom.

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS NEGERI SURABAYA

2025

Daftar Isi

BAB I.....	1
PENDAHULUAN	1
1.1 Latar Belakang.....	1
1.2 Rumusan Masalah	1
1.3 Tujuan Penelitian	1
1.4 Manfaat Penelitian.....	2
1.5 Batasan Masalah	2
BAB II.....	3
TINJAUAN PUSTAKA	3
2.1 Pengolahan Citra Digital	3
2.2 Noise Pada Citra.....	3
2.3 Median Filtering.....	3
2.4 Operasi Morfologi (Erosi dan Dilasi) dan Opening	3
2.5 Evaluasi Kualitas Citra: MSE & PSNR.....	4
2.6 Tinjauan Penelitian Terkait.....	5
BAB III.....	6
METODOLOGI PENELITIAN	6
3.1 Alur Sistem.....	6
3.2 Dataset.....	7
3.3 Perancangan Sistem	8
3.4 Tahapan Implementasi.....	10
BAB IV	13
HASIL DAN PEMBAHASAN	13
4.1 Platform Pengembangan	13
4.2 Implementasi Kode Program	13
4.3 Hasil Pengujian.....	22
4.4 Evaluasi dan Pembahasan	25
BAB V	29
PENUTUP.....	29
5.1 Kesimpulan	29
5.2 Saran.....	29
DAFTAR PUSTAKA	30

BAB I

PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi digital saat ini memberikan kemudahan dalam proses dokumentasi dan digitalisasi ini dapat dilakukan melalui perangkat scanner atau kamera digital, yang mengubah data fisik menjadi citra digital. Namun, proses ini seringkali menimbulkan permasalahan dalam hal kualitas hasil gambar, khususnya munculnya noise pada citra, seperti bintik-bintik hitam atau putih yang mengganggu struktur informasi visual. Noise dapat disebabkan oleh kualitas kertas, kotoran pada dokumen, pencahayaan yang tidak merata, maupun kekurangan perangkat pemindai.

Noise pada citra digital mengakibatkan informasi visual menjadi kurang jelas, bahkan dapat mengganggu tahap selanjutnya dalam pemrosesan citra seperti pengenalan karakter atau segmentasi objek. Oleh karena itu, dibutuhkan metode untuk mengurangi atau menghilangkan noise tersebut agar kualitas citra dapat meningkat. Salah satu pendekatan yang banyak digunakan dalam pengolahan citra digital adalah penggunaan filter dan operasi morfologi.

Metode Median Filtering merupakan salah satu Teknik yang sangat populer dalam reduksi noise, terutama jenis Salt & Pepper Noise. Teknik ini tergolong sebagai filter non-linear, di mana nilai piksel pusat diganti dengan nilai median dari tetangganya dalam kernel (3x3, 5x5, 7x7, dst).

Selain itu, metode Morphological Opening yang merupakan gabungan dari operasi erosi dan dilasi, juga banyak digunakan untuk mereduksi noise pada citra biner. Menurut Sunil Bhutada (2022), Metode ini efektif untuk menghilangkan objek kecil tanpa merusak bentuk utama citra.

Dalam jurnal oleh Amru Yasir dkk (2023), kedua metode tersebut telah diterapkan untuk memperbaiki citra manuskrip yang rusak akibat noise. Hasilnya menunjukkan bahwa Median Filtering memberikan nilai MSE lebih kecil dan PSNR lebih tinggi, sehingga lebih efektif dibandingkan Morphological Opening.

Penelitian ini bertujuan untuk mengimplementasikan dan membandingkan kedua metode tersebut menggunakan aplikasi berbasis Python dengan tampilan GUI, guna mengetahui metode mana yang paling optimal dalam mereduksi noise citra digital.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan, rumusan masalah dalam penelitian ini adalah sebagai berikut:

1. Bagaimana cara mengimplementasikan metode Median Filtering dan Morphological Opening untuk mereduksi noise pada citra digital hasil pemindaian?
2. Metode manakah yang lebih efektif dalam mereduksi noise berdasarkan nilai MSE (Mean Square Error) dan PSNR (Peak Signal to Noise Ratio)?

1.3 Tujuan Penelitian

Penelitian ini bertujuan untuk:

1. mengimplementasikan metode Median Filtering dan Morphological Opening dalam proses reduksi noise pada citra digital.
2. Membandingkan efektivitas kedua metode berdasarkan hasil evaluasi menggunakan nilai MSE dan PSNR.
3. Mengembangkan aplikasi berbasis GUI untuk memudahkan proses pengolahan dan evaluasi citra digital.

1.4 Manfaat Penelitian

Adapun manfaat yang dapat diperoleh dari penelitian ini adalah:

1. memberikan pemahaman dan penerapan praktis mengenai metode Median Filtering dan morphological Opening dalam pengolahan citra digital.
2. Menjadi referensi dalam pengembangan sistem reduksi noise pada citra hasil digitalisasi dokumen, khususnya manuskrip.
3. Menyediakan aplikasi sederhana berbasis GUI yang dapat digunakan untuk menguji dan membandingkan hasil reduksi noise secara langsung.

1.5 Batasan Masalah

Agar penelitian lebih terarah, Batasan masalah dalam penelitian ini adalah sebagai berikut:

1. Citra yang digunakan berupa gambar manuskrip berformat .jpg hasil pemindaian atau kamera.
2. Jenis noise yang difokuskan adalah Salt & Pepper.
3. Metode yang digunakan untuk reduksi noise terbatas pada Median Filtering dan Morphological Opening.
4. Proses Evaluasi dilakukan dengan menggunakan metrik MSE (Mean Square Error) dan PSNR (Peak Signal to Noise Ratio)
5. Implementasi dilakukan menggunakan Bahasa pemrograman Python dan antarmuka berbasis GUI dengan Library Tkinter.

BAB 11

TINJAUAN PUSTAKA

2.1 Pengolahan Citra Digital

Pengolahan citra digital (Digital Image Processing) merupakan proses manipulasi citra dalam bentuk digital menggunakan algoritma komputer untuk meningkatkan kualitas visual atau mengekstraksi informasi penting dari citra tersebut. Proses ini mencakup berbagai tahapan, mulai dari akuisisi citra, pra-pemrosesan, segmentasi, hingga analisis dan interpretasi data citra. Keunggulan pengolahan citra digital dibandingkan dengan pengolahan citra analog terletak pada fleksibilitasnya dalam menerapkan berbagai algoritma kompleks serta kemampuannya dalam menghindari akumulasi noise dan distorsi sinyal selama pemrosesan.

2.2 Noise Pada Citra

Noise pada citra digital merupakan gangguan acak yang dapat menurunkan kualitas visual dan mengganggu proses analisis citra. Noise dapat muncul akibat berbagai faktor, seperti gangguan sinyal selama akuisisi, pencahayaan yang tidak merata, atau kesalahan transmisi data digital. Beberapa jenis noise yang umum ditemukan dalam pengolahan citra antara lain:

- Salt and Pepper Noise: ditandai dengan kemunculan titik-titik hitam dan putih secara acak pada citra, seringkali disebabkan oleh gangguan impulsif pada sensor.
- Gaussian Noise: memiliki distribusi normal dan menyebabkan perubahan nilai intensitas piksel secara halus, biasanya muncul karena fluktuasi sensor atau elektronik.
- Speckle Noise: biasanya terdapat pada citra hasil pemrosesan koheren seperti ultrasound dan radar, berbentuk bintik-bintik granular.

Kehadiran noise pada citra dapat mempersulit proses analisis lanjutan seperti deteksi tepi, segmentasi objek, atau pengenalan pola. Oleh karena itu, diperlukan teknik reduksi noise yang efektif guna meningkatkan kualitas citra secara menyeluruh.

2.3 Median Filtering

Median filtering adalah metode penyaringan non-linier yang umum digunakan dalam pengolahan citra digital untuk menghilangkan noise, khususnya jenis salt and pepper. Teknik ini bekerja dengan mengganti nilai suatu piksel dengan nilai median dari lingkungan sekitarnya yang ditentukan oleh ukuran kernel (misalnya 3×3 , 5×5 , atau 7×7). Berbeda dengan filter linier seperti mean filtering, median filter memiliki keunggulan dalam mempertahankan tepi objek karena tidak memperhitungkan nilai rata-rata, melainkan nilai tengah (median). Hal ini membuat median filter sangat efektif dalam menghapus noise impulsif tanpa mengaburkan detail citra yang penting.

Efektivitas median filter sangat bergantung pada ukuran kernel yang digunakan. Ukuran kernel yang kecil cenderung mempertahankan lebih banyak detail, sedangkan kernel yang lebih besar lebih efektif dalam mereduksi noise tetapi berpotensi menghilangkan informasi penting dari citra.

2.4 Operasi Morfologi (Erosi dan Dilasi) dan Opening

Operasi morfologi merupakan teknik pengolahan citra yang berbasis pada teori himpunan dan bentuk geometris objek dalam citra, biasanya diterapkan pada citra biner atau grayscale. Operasi ini menggunakan elemen struktur (structuring element) untuk memanipulasi bentuk objek berdasarkan karakteristik geometri lokal.

- Erosi adalah operasi morfologi yang bertujuan untuk mengikis batas objek dalam citra. Erosi akan mengurangi ukuran objek terang dan menghilangkan piksel-piksel kecil yang terisolasi. Operasi ini secara umum dapat menghilangkan noise berukuran kecil dan memisahkan objek-objek yang saling berdekatan.
- Dilasi merupakan operasi kebalikan dari erosi, yaitu memperbesar ukuran objek terang dalam citra. Proses ini akan menambahkan piksel di sekitar batas objek, mengisi lubang kecil, dan menyatukan bagian-bagian objek yang terpisah.
- Opening adalah kombinasi dari operasi erosi yang diikuti oleh dilasi. Tujuan dari opening adalah untuk menghilangkan objek kecil atau noise tanpa mempengaruhi bentuk dan ukuran objek utama. Opening sangat efektif dalam membersihkan noise pada citra biner atau grayscale, serta dalam mempertahankan struktur utama dari objek yang lebih besar daripada elemen struktur.

Pemilihan ukuran dan bentuk structuring element berpengaruh besar terhadap hasil akhir operasi morfologi. Oleh karena itu, pemilihan kernel yang tepat menjadi bagian penting dari proses pengolahan morfologi.

2.5 Evaluasi Kualitas Citra: MSE & PSNR

Evaluasi kualitas citra hasil pemrosesan dapat dilakukan menggunakan parameter kuantitatif seperti Mean Squared Error (MSE) dan Peak Signal to Noise Ratio (PSNR). Kedua metrik ini sering digunakan untuk membandingkan citra hasil pemrosesan terhadap citra referensi (ground truth).

- Mean Squared Error (MSE)
MSE adalah ukuran rata-rata kesalahan kuadrat antara piksel-piksel pada citra asli dan citra hasil pemrosesan. Nilai MSE yang lebih kecil menunjukkan kesalahan yang lebih rendah, sehingga hasil pemrosesan dianggap lebih baik.
Rumus MSE adalah sebagai berikut:

$$MSE = \frac{1}{mn} \sum_{i=1}^m \sum_{j=1}^n [I(i,j) - K(i,j)]^2$$

Dimana:

$I(i,j)$: piksel pada citra asli

$K(i,j)$: piksel pada citra hasil pemrosesan

$m \times n$: ukuran dimensi citra

- Peak Signal to Noise Ratio (PSNR)
PSNR merupakan ukuran rasio antara sinyal maksimum dengan noise yang mengganggu. PSNR dinyatakan dalam satuan desibel (dB), dan semakin tinggi nilai PSNR, maka kualitas citra dianggap semakin baik.
Rumus PSNR:

$$PSNR = 10 \cdot \log_{10} \left(\frac{MAX^2}{MSE} \right)$$

Di mana MAX adalah nilai maksimum intensitas piksel, biasanya 255 untuk citra 8-bit. Nilai PSNR di atas 30 dB umumnya dianggap baik dalam pengolahan citra .

2.6 Tinjauan Penelitian Terkait

Penelitian yang dilakukan oleh Amru Yasir, Welnof Satria, dan Putri Yuanda (2023) dalam jurnal berjudul "Digital Image Processing: Metode Median Filtering dan Morfologi Opening dalam Reduksi Noise Citra" membahas mengenai penerapan metode Median Filtering dan Morfologi Opening untuk mereduksi noise pada citra hasil pemindaian. Dalam penelitian tersebut dijelaskan bahwa hasil citra digital dari proses pemindaian dokumen seringkali mengalami gangguan berupa derau atau noise, terutama karena kualitas kertas yang buruk atau kotoran pada dokumen. Gangguan ini dapat menyebabkan citra menjadi tidak jelas dan mengganggu proses interpretasi visual.

Penulis menerapkan dua metode utama untuk menangani permasalahan tersebut. Pertama adalah Median Filtering, yaitu metode penyaringan non-linear yang bekerja dengan mengganti nilai piksel tengah dalam jendela yang telah diurutkan, sehingga efektif dalam menghilangkan noise seperti salt and pepper. Metode ini terbukti dapat mempertahankan tepi objek citra dengan baik dibandingkan metode mean filter yang cenderung mengaburkan tepi.

Metode kedua yang digunakan adalah morfologi opening, yaitu salah satu operasi morfologi yang dilakukan setelah proses erosi dan diikuti dengan dilasi. Teknik ini efektif dalam menghilangkan noise kecil dan objek tidak diinginkan pada citra biner, serta mempertahankan bentuk asli objek utama pada citra. Dalam konteks citra dokumen, operasi ini membantu membersihkan titik-titik noise yang tidak relevan dan meningkatkan keterbacaan teks atau konten pada dokumen.

Penelitian ini menunjukkan bahwa kombinasi antara Median Filtering dan Morfologi Opening mampu meningkatkan kualitas visual dari citra digital hasil pemindaian. Hasil dari penelitian ini memberikan dasar pendekatan yang relevan untuk pengolahan citra digital khususnya dalam konteks perbaikan kualitas dokumen hasil pemindaian, serta memberikan kontribusi terhadap pengembangan metode yang efisien dan efektif dalam bidang pengolahan citra.

BAB III

METODOLOGI PENELITIAN

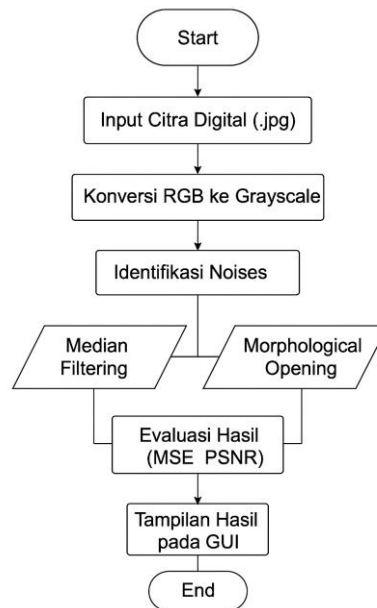
3.1 Alur Sistem

Alur sistem pada penelitian ini mengikuti tahapan-tahapan yang telah dijelaskan dalam jurnal, yaitu proses reduksi noise pada citra digital menggunakan dua metode: Median Filtering dan Morphological Opening.

Adapun alur sistem secara umum terdiri dari Langkah-langkah berikut:

1. Input Citra
Citra yang digunakan merupakan gambar manuskrip dengan format .jpg yang dihasilkan dari pemindaian atau foto dokumen/manuskrip
2. Konversi RGB ke Greyscale
Citra warna dikonversi menjadi greyscale agar proses binarisasi dan pemrosesan citra lebih sederhana dan efisien.
3. Binarisasi Citra (Thresholding)
Citra greyscale diubah menjadi citra hitam-putih (biner) menggunakan metode ambang otomatis, yaitu threshold Otsu atau metode lain berbasis histogram.
4. Identifikasi Noise
Setelah citra biner terbentuk, dilakukan identifikasi noise dengan memeriksa piksel yang berbeda signifikan dari tetangga sekitarnya. Titik-titik yang terdeteksi sebagai noise ditandai secara visual (misalnya dengan warna abu-abu/128).
5. Reduksi Noise dengan Dua Metode:
 - Median Filtering: Nilai piksel pusat diganti dengan median dari nilai tetangganya dalam window ukuran 3x3, 5x5, atau 7x7.
 - Morphological Opening: Diterapkan melalui kombinasi erosi lalu dilasi dengan window yang sama. Operasi ini efektif menghilangkan noise kecil tanpa merusak bentuk utama objek citra.
6. Evaluasi Hasil
Hasil dari kedua metode dibandingkan menggunakan:
 - MSE (Mean Square Error): Mengukur rata-rata error kuadrat antara citra asli dan citra hasil reduksi.
 - PSNR (Peak Signal to Noise Ratio): Mengukur rasio sinyal terhadap noise, di mana nilai lebih tinggi menunjukkan hasil yang lebih baik.
7. Tampilan Hasil pada GUI
Semua hasil proses ditampilkan pada antarmuka:
 - Visualisasi citra (asli, grayscale, biner, identifikasi noise, hasil median, hasil opening)
 - Nilai evaluasi MSE dan PSNR dari masing-masing metode
 - Kesimpulan otomatis mengenai metode yang lebih efektif.

Berikut merupakan Flowchart dari alur sistem di atas:



3.2 Dataset

Dataset yang digunakan dalam penelitian ini berupa sebuah citra digital dengan format file .jpg yang mengandung noise. Citra tersebut merupakan gambar manuskrip atau dokumen yang telah mengalami gangguan visual seperti bintik-bintik atau bercak (umumnya berupa Salt & Pepper Noise) akibat proses pemindaian atau pengambilan gambar dengan kamera.

Citra tersebut digunakan sebagai objek pengujian utama untuk penerapan dua metode reduksi noise, yaitu Median Filtering dan Morphological Opening. Proses pengujian dilakukan secara berulang dengan variasi ukuran window atau karnel filter (3x3, 5x5, dan 7x7) untuk membandingkan efektivitas masing-masing metode.

Karakteristik citra:

- Format: .jpg
- Warna awal: RGB
- Ukuran: Disesuaikan
- Jenis Noise: Dominan Salt & Pepper Noise

Berikut adalah contoh dataset citra biner yang digunakan:



3.3 Perancangan Sistem

Perancangan sistem dalam penelitian ini bertujuan untuk membangun sebuah aplikasi sederhana berbasis GUI (*Graphical User Interface*) menggunakan bahasa pemrograman Python. Aplikasi ini digunakan untuk memproses citra digital yang mengandung noise, serta membandingkan hasil reduksi noise menggunakan dua metode: Median Filtering dan Morphological Opening.

Adapun komponen-komponen utama dalam perancangan sistem adalah sebagai berikut:

1. Antarmuka Pengguna (GUI)

Sistem dirancang menggunakan library Tkinter pada python untuk membuat tampilan antarmuka sederhana yang terdiri dari:

- Tombol “Open Image”: untuk memilih citra .jpg dari computer.
- Combo box: untuk memilih ukuran window filter (3x3, 5x5, atau 7x7).
- Tombol “Process”: untuk memulai proses binarisasi, identifikasi noise, reduksi noise, dan evaluasi.
- Text box hasil: menampilkan nilai MSE dan PSNR dari kedua metode.
- Label Hasil: menampilkan citra hasil masing-masing metode secara visual.

2. Komponen Proses Utama

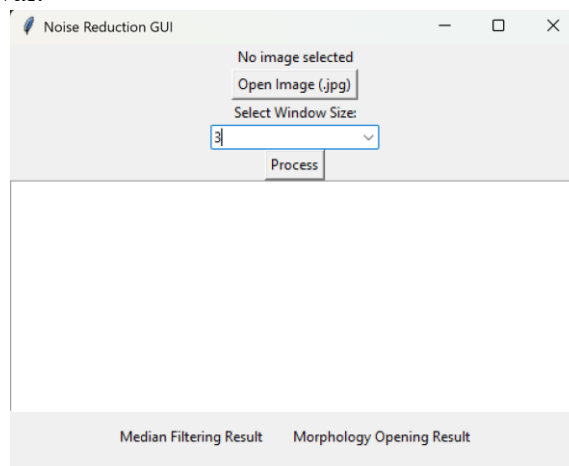
Berikut adalah tahapan proses yang dirancang di dalam sistem:

- Input Citra: Membaca filr .jpg
- Konversi RGB ke Grayscale: Mengubah citra warna menjadi keabuan.
- Binarisasi: Mengubah citra grayscale menjadi citra biner (hitam-putih).
- Identifikasi Noise: menandai piksel yang berbeda dari tetangga sebagai noise.
- Proses Median Filtering: Menerapkan filter median pada citra biner.
- Proses Morphological Opening: Menerapkan operasi morfologi (erosi lalu dilasi)
- Evaluasi kualitas: Menghitung MSE dan PSNR dari hasil masing-masing metode.
- Tampilan Output: Menampilkan hasil evaluasi dan citra hasil ke dalam GUI.

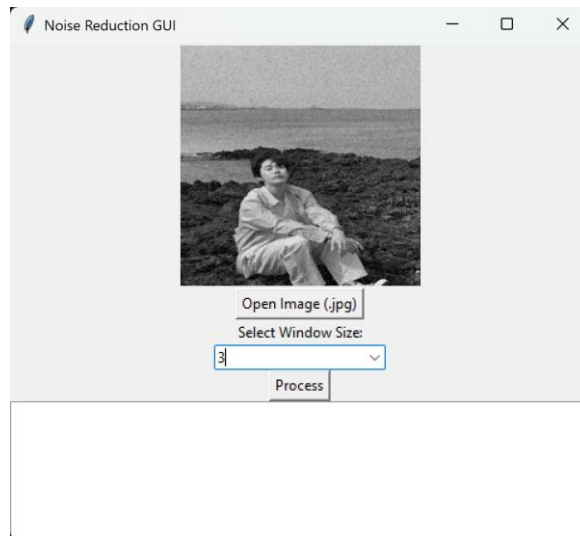
3. Diagram Antarmuka

Berikut lampiran diagram GUI pada aplikasi:

- Tampilan Awal:



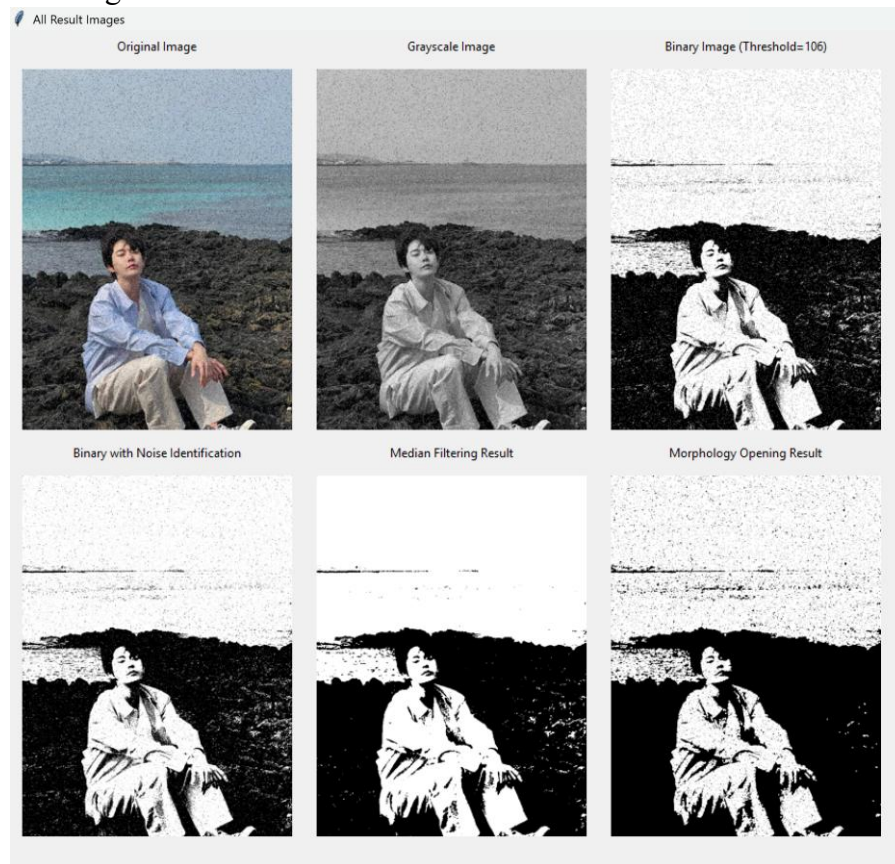
- Tampilan setelah memuat citra:



- Tampilan setelah menekan tombol “Process” yang menampilkan hasil dan evaluasi:



Dan berikut merupakan hasil dari tahapan process yang ditampilkan dalam bentuk gambar:



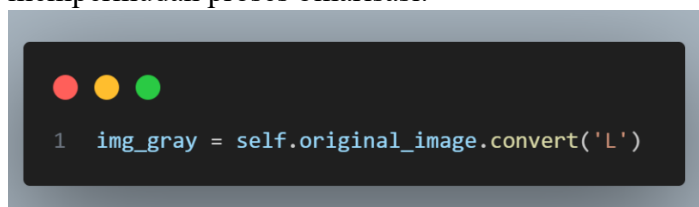
3.4 Tahapan Implementasi

Implementasi sistem dilakukan dalam beberapa tahap, dimulai dari proses input citra hingga evaluasi hasil pengolahan. Semua proses dijalankan dalam satu aplikasi berbasis Python menggunakan Tkinter, PIL, dan NumPy.

Berikut adalah tahapan implementasi secara rinci:

1. Input dan Konversi Citra

Pengguna memilih citra digital dengan format .jpg menggunakan tombol “Open Image”. Kemudian citra yang dimuat akan dikonversi dari RGB ke grayscale untuk mempermudah proses binarisasi:



2. Binarisasi Citra

Grayscale dikonversi menjadi citra biner (hitam-putih) menggunakan metode threshold otomatis berdasarkan histogram.

```
1 binary_img, threshold = binarize(self.image_array)
```

3. Identifikasi Noise

Piksel-piksel yang dianggap berbeda dari tetangganya (outlier) diidentifikasi sebagai noise. Piksel ini ditandai untuk analisis.

```
1 noise_identified = identify_noise(binary_img)
```

4. Reduksi Noise

Dilakukan dengan dua metode secara paralel:

- Median Filtering: Mengganti nilai setiap piksel dengan nilai median dari tetangga dalam window 3x3, 5x5, 7x7.

```
1 median_result = median_filter(binary_img, k)
```

- Morphological Opening: Menggunakan kombinasi operasi erosi dan dilasi dengan ukuran kernel yang sama.

```
1 morph_result = morphology_opening(binary_img, k)
```

5. Evaluasi kualitas hasil

Setiap hasil dari dua metode dihitung nilai MSE dan PSNR-nya untuk mengetahui efektivitas masing-masing.

```
1 mse_median = mse(self.ref_array, median_result)
2 psnr_median = psnr(self.ref_array, median_result)
3
4 mse_morph = mse(self.ref_array, morph_result)
5 psnr_morph = psnr(self.ref_array, morph_result)
```

6. Tampilan Hasil pada GUI

- Citra hasil median filtering dan morphological opening ditampilkan pada label.
- Nilai MSE dan PSNR dicetak ke dalam textbox.
- Terdapat juga tampilan semua citra proses di window tambahan.

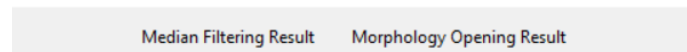
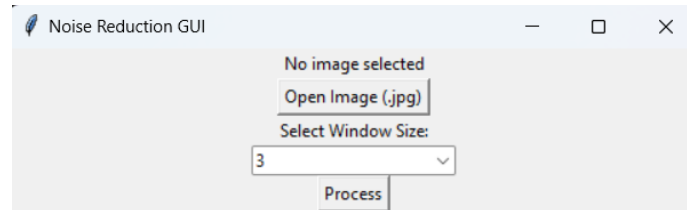
Tahapan ini membentuk keseluruhan siklus kerja aplikasi mulai dari input hingga evaluasi hasil, sesuai dengan alur sistem yang telah dirancang dan flowchart yang telah dibuat sebelumnya.

BAB IV

HASIL DAN PEMBAHASAN

4.1 Platform Pengembangan

Aplikasi ini dikembangkan menggunakan bahasa pemrograman Python dengan pustaka utama Tkinter untuk antarmuka grafis (GUI), NumPy untuk komputasi numerik, dan Pillow (PIL) untuk pemrosesan gambar. Program diuji pada sistem operasi Windows 10 dengan spesifikasi perangkat keras Ryzen 5 dan RAM 8 GB. Editor pengembangan yang digunakan adalah Visual Studio Code.



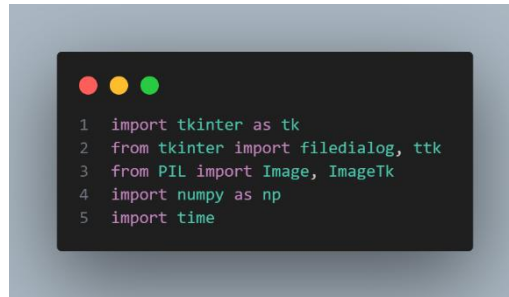
4.2 Implementasi Kode Program

Kode program dibangun dalam satu file utama dengan struktur sebagai berikut:

- Fungsi binarize digunakan untuk mengubah citra grayscale menjadi citra biner dengan metode Otsu Thresholding.
- Fungsi identify_noise digunakan untuk mendeteksi noise berupa piksel yang berbeda dari semua tetangganya. Piksel noise ditandai dengan warna abu-abu (nilai 128).
- Fungsi median_filter dan morphology_opening digunakan untuk melakukan reduksi noise. Median filter bekerja dengan mengganti setiap piksel dengan median dari tetangganya, sedangkan morphology opening menggunakan kombinasi erosi dan dilasi.
- Metode evaluasi seperti MSE (Mean Squared Error) dan PSNR (Peak Signal-to-Noise Ratio) digunakan untuk menilai efektivitas metode reduksi noise.
- Kelas NoiseReductionGUI membangun antarmuka pengguna dengan tombol input gambar, pilihan ukuran kernel (3, 5, 7), serta area hasil evaluasi.
- Tampilan GUI juga menyediakan jendela tambahan yang menampilkan citra hasil transformasi secara visual: gambar asli, grayscale, biner, noise, hasil median, dan hasil morfologi.

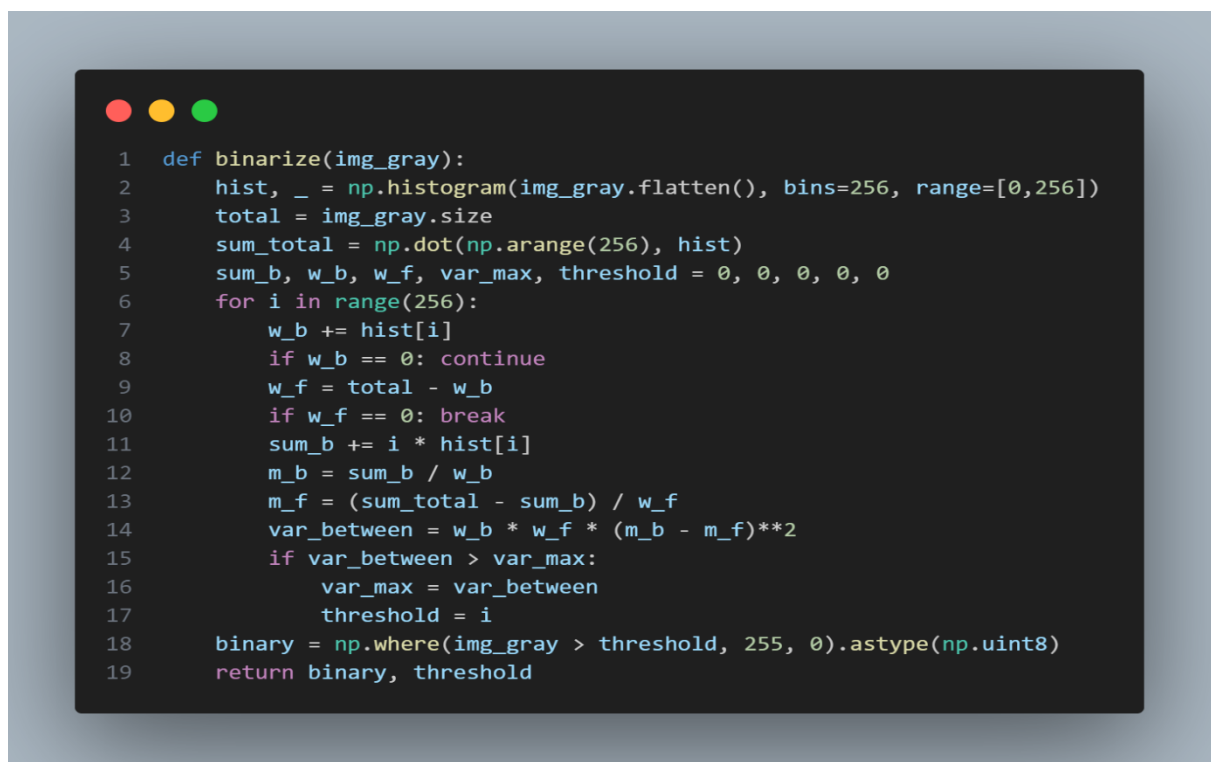
Keseluruhan kode yaitu sebagai berikut:

- a. Library



Program ini memakai beberapa library penting: tkinter digunakan untuk membuat tampilan antarmuka pengguna agar bisa membuka file gambar, memilih opsi, dan melihat hasilnya secara interaktif; filedialog dari tkinter membantu pengguna memilih file gambar dari komputer; ttk menyediakan widget yang lebih modern seperti combobox untuk memilih ukuran kernel; Pillow (PIL) berfungsi untuk membuka, mengubah ukuran, dan mengonversi gambar ke format yang bisa diproses, terutama ke grayscale; numpy digunakan untuk manipulasi array gambar, seperti padding, filtering, dan perhitungan nilai statistik; serta time dipakai untuk mengukur lama proses filtering agar bisa dibandingkan performanya.

b. Binarize



Fungsi ini digunakan untuk mengubah gambar grayscale menjadi citra biner (hitam-putih) dengan menentukan ambang batas (threshold) optimal menggunakan metode Otsu.

Prosesnya dimulai dengan menghitung histogram dari seluruh pixel dalam gambar grayscale, yang merepresentasikan frekuensi kemunculan nilai intensitas dari 0 hingga 255. Selanjutnya, fungsi ini mencari ambang batas (threshold) yang memaksimalkan

variansi antar kelas—yakni variansi antara kelompok piksel gelap dan terang. Ambang batas yang memberikan nilai variansi terbesar dipilih sebagai threshold optimal.

Setelah threshold ditemukan, setiap pixel diubah menjadi 255 (putih) jika nilai intensitasnya lebih tinggi dari threshold tersebut, atau menjadi 0 (hitam) jika sebaliknya. Hasilnya adalah gambar biner yang siap diproses lebih lanjut.

c. Identifikasi Noise

```
1 def identify_noise(binary_img):
2     """Tandai pixel noise: pixel yang berbeda dengan semua 8 tetangga"""
3     padded = np.pad(binary_img, 1, mode='edge')
4     noise_img = binary_img.copy()
5     rows, cols = binary_img.shape
6     for i in range(rows):
7         for j in range(cols):
8             center = padded[i+1, j+1]
9             neighbors = padded[i:i+3, j:j+3].flatten()
10            neighbors = np.delete(neighbors, 4) # hapus pixel center sendiri
11            if all(center != neighbors):
12                noise_img[i, j] = 128 # tandai noise dengan abu-abu tengah
13    return noise_img
```

Fungsi ini bertugas mengidentifikasi noise dalam gambar biner dengan pendekatan sederhana tapi intuitif. Idennya adalah mendeteksi pixel yang berbeda dari semua tetangganya dalam 8 arah (atas, bawah, kiri, kanan, dan diagonal).

Pertama, citra diberi padding (bingkai tambahan) agar bisa membaca tetangga tanpa indeks error di tepi gambar. Kemudian setiap pixel diperiksa: jika pixel tersebut memiliki nilai berbeda dari seluruh 8 tetangganya, maka dianggap sebagai noise. Pixel yang diidentifikasi sebagai noise ditandai dengan nilai 128 (abu-abu).

Output fungsi ini adalah gambar biner yang sama, tetapi dengan piksel noise ditandai warna abu-abu untuk visualisasi.

d. Median Filter

```

1 def median_filter(img, k=3):
2     pad = k // 2
3     padded = np.pad(img, pad, mode='edge')
4     output = np.zeros_like(img)
5     for i in range(img.shape[0]):
6         for j in range(img.shape[1]):
7             output[i, j] = np.median(padded[i:i+k, j:j+k])
8     return output

```

Ini adalah implementasi manual dari metode Median Filtering, salah satu metode paling populer untuk mengurangi noise salt-and-pepper pada citra.

Fungsi ini memproses citra dengan mengambil jendela (window) berukuran $k \times k$ di sekitar setiap pixel, lalu mengganti pixel pusat dengan nilai median dari semua pixel dalam jendela tersebut. Dengan begitu, nilai ekstrem yang mengganggu akan dihapus atau dikurangi.

Citra awal dipad dengan teknik padding (edge padding), kemudian setiap jendela di-scan satu per satu untuk mendapatkan nilai median lokalnya. Hasilnya adalah gambar yang lebih bersih dari noise acak.

e. Erosi

```

1 def erosion(img, k=3):
2     pad = k // 2
3     padded = np.pad(img, pad, mode='constant', constant_values=255)
4     output = np.zeros_like(img)
5     for i in range(img.shape[0]):
6         for j in range(img.shape[1]):
7             output[i, j] = np.min(padded[i:i+k, j:j+k])
8     return output

```

Fungsi ini melakukan operasi morfologi erosi, yang umum digunakan dalam pemrosesan citra biner. Tujuannya adalah mengecilkan objek putih dan menghilangkan noise kecil.

Setiap pixel diproses dengan mengambil nilai minimum dari seluruh pixel dalam jendela $k \times k$ di sekelilingnya. Karena bekerja pada citra biner, nilai minimum biasanya

menyebabkan penyusutan objek terang dan menghapus titik-titik terang yang terisolasi (noise).

f. Dilatasi

```
1 def dilation(img, k=3):
2     pad = k // 2
3     padded = np.pad(img, pad, mode='constant', constant_values=0)
4     output = np.zeros_like(img)
5     for i in range(img.shape[0]):
6         for j in range(img.shape[1]):
7             output[i, j] = np.max(padded[i:i+k, j:j+k])
8     return output
```

Kebalikan dari erosion, fungsi ini melakukan dilation, yang memperbesar area terang (putih) pada gambar. Dalam proses ini, setiap pixel akan digantikan oleh nilai maksimum dari jendela $k \times k$ di sekitarnya.

Dilation digunakan untuk menutupi celah kecil dan menebalkan garis-garis yang tipis dalam citra biner. Biasanya digunakan setelah erosion sebagai bagian dari teknik opening.

g. Morfologi Opening

```
1 def morphology_opening(img, k=3):
2     return dilation(erosion(img, k), k)
```

Fungsi ini menggabungkan dua operasi morfologi sebelumnya: erosion diikuti dengan dilation. Kombinasi ini disebut opening.

Metode ini ideal untuk menghilangkan noise kecil pada citra biner tanpa terlalu merusak struktur utama objek dalam gambar. Opening menyaring noise yang kecil (biasanya titik terang acak) sambil menjaga bentuk asli objek.

h. MSE



```

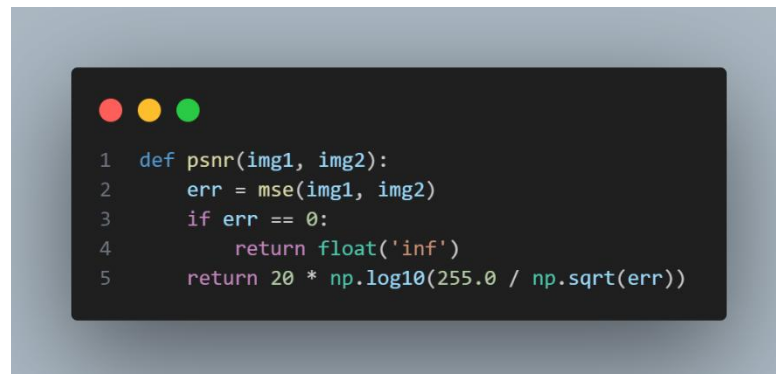
1 def mse(img1, img2):
2     return np.mean((img1.astype(float) - img2.astype(float)) ** 2)

```

Fungsi ini menghitung Mean Squared Error (MSE) antara dua gambar, yaitu jumlah kuadrat perbedaan antara pixel gambar pertama dan kedua, lalu dirata-ratakan.

MSE digunakan sebagai ukuran objektif seberapa berbeda dua gambar—semakin kecil nilainya, semakin mirip gambar tersebut.

i. PNSR



```

1 def psnr(img1, img2):
2     err = mse(img1, img2)
3     if err == 0:
4         return float('inf')
5     return 20 * np.log10(255.0 / np.sqrt(err))

```

Fungsi ini menghitung Peak Signal-to-Noise Ratio (PSNR) dari dua gambar. PSNR adalah ukuran kuantitatif yang sering digunakan untuk menilai kualitas hasil kompresi atau pemrosesan citra.

Nilainya dihitung dari MSE yang telah didapat sebelumnya. PSNR lebih tinggi berarti kualitas gambar hasil pemrosesan lebih baik (lebih mirip dengan gambar asli).

j. GUI

Kelas ini membangun Graphical User Interface (GUI) untuk proses reduksi noise menggunakan pustaka tkinter.

Berikut elemen-elemennya secara naratif:

- a. Tampilan awal GUI menampilkan tombol untuk membuka file gambar .jpg, label untuk menunjukkan gambar yang dipilih, dan pilihan ukuran jendela (kernel) untuk pemrosesan.

```

1 class NoiseReductionGUI:
2     def __init__(self, root):
3         self.root = root
4         self.root.title("Noise Reduction GUI")
5
6         self.image_label = tk.Label(root, text="No image selected")
7         self.image_label.pack()
8
9         self.open_btn = tk.Button(root, text="Open Image (.jpg)", command=self.open_image)
10        self.open_btn.pack()
11
12        self.kernel_label = tk.Label(root, text="Select Window Size:")
13        self.kernel_label.pack()
14
15        self.kernel_combo = ttk.Combobox(root, values=[3, 5, 7])
16        self.kernel_combo.current(0)
17        self.kernel_combo.pack()
18
19        self.process_btn = tk.Button(root, text="Process", command=self.process_image)
20        self.process_btn.pack()
21
22        self.result_text = tk.Text(root, height=12, width=60)
23        self.result_text.pack()
24
25        self.frame_results = tk.Frame(root)
26        self.frame_results.pack()
27
28        self.label_median = tk.Label(self.frame_results, text="Median Filtering Result")
29        self.label_median.grid(row=0, column=0, padx=10, pady=10)
30
31        self.label_morph = tk.Label(self.frame_results, text="Morphology Opening Result")
32        self.label_morph.grid(row=0, column

```

```

1  def open_image(self):
2      path = filedialog.askopenfilename(filetypes=[("JPG Files", "*.jpg")])
3      if path:
4          self.original_image = Image.open(path)
5          img_gray = self.original_image.convert('L')
6          self.image_array = np.array(img_gray)
7          img_thumbnail = img_gray.resize((200, 200))
8          self.tk_img_input = ImageTk.PhotoImage(img_thumbnail)
9          self.image_label.config(image=self.tk_img_input, text="")
10
11         self.ref_array = self.image_array.copy()
12
13     def tampilkan_hasil(self, array, label):
14         img = Image.fromarray(array.astype('uint8'))
15         img = img.resize((200, 200))
16         tk_img = ImageTk.PhotoImage(img)
17         label.config(image=tk_img)
18         label.image = tk_img
19
20     def process_image(self):
21         if self.image_array is None:
22             return
23
24         k = int(self.kernel_combo.get())
25         binary_img, threshold = binarize(self.image_array)
26
27         noise_identified = identify_noise(binary_img)
28
29         start1 = time.time()
30         median_result = median_filter(binary_img, k)
31         end1 = time.time()
32
33         start2 = time.time()
34         morph_result = morphology_opening(binary_img, k)
35         end2 = time.time()
36

```

- b. Saat pengguna mengklik "Open Image", gambar akan dimuat, dikonversi ke grayscale, lalu ditampilkan dalam ukuran kecil.
- c. Setelah itu, saat pengguna menekan tombol "Process", proses berikut terjadi secara bertahap:
 - Gambar diubah ke citra biner.
 - Noise diidentifikasi dan ditandai.
 - Dua metode reduksi noise dilakukan: median filter dan morphology opening.
 - MSE dan PSNR dihitung untuk membandingkan kualitas hasilnya.
 - Waktu proses juga diukur dan ditampilkan.
- d. GUI juga menampilkan hasil citra dari kedua metode dalam bentuk gambar, serta membuka jendela baru yang menampilkan semua tahapan pemrosesan secara visual, termasuk citra asli, grayscale, biner, hasil identifikasi noise, dan hasil kedua metode filtering.

k. New Frame

```

1  def show_result_window(self, binary_img, noise_img, median_result, morph_result, threshold):
2      win = tk.Toplevel(self.root)
3      win.title("All Result Images")
4
5      images = [
6          (self.original_image, "Original Image"),
7          (self.original_image.convert('L'), "Grayscale Image"),
8          (Image.fromarray(binary_img), f"Binary Image (Threshold={threshold})"),
9          (Image.fromarray(noise_img), "Binary with Noise Identification"),
10         (Image.fromarray(median_result), "Median Filtering Result"),
11         (Image.fromarray(morph_result), "Morphology Opening Result"),
12     ]
13
14     self.result_labels = []
15     self.result_pil_images = []
16     self.result_tk_images = []
17
18     for idx, (img, title) in enumerate(images):
19         row = idx // 3
20         col = idx % 3
21         tk.Label(win, text=title).grid(row=row*2, column=col, padx=10, pady=5)
22
23         lbl = tk.Label(win)
24         lbl.grid(row=row*2 + 1, column=col, padx=10, pady=5)
25         self.result_labels.append(lbl)
26         self.result_pil_images.append(img)
27
28     def resize_and_update(event=None):
29         win.update_idletasks()
30         max_width, max_height = 600, 600
31
32         width = max((win.winfo_width() - 60) // 3, 50)
33         height = max((win.winfo_height() - 120) // 2, 70)
34
35         width = min(width, max_width)
36         height = min(height, max_height)
37
38         self.result_tk_images.clear()
39         for i, pil_img in enumerate(self.result_pil_images):
40             pil_w, pil_h = pil_img.size
41             ratio = min(width / pil_w, height / pil_h)
42             new_w = int(pil_w * ratio)
43             new_h = int(pil_h * ratio)
44
45             img_resized = pil_img.resize((new_w, new_h))
46             tk_img = ImageTk.PhotoImage(img_resized)
47             self.result_tk_images.append(tk_img)
48             self.result_labels[i].config(image=tk_img)
49             self.result_labels[i].image = tk_img
50
51         win.bind("<Configure>", resize_and_update)
52         win.update_idletasks()
53         resize_and_update()
54
55     if __name__ == "__main__":
56         root = tk.Tk()
57         app = NoiseReductionGUI(root)
58         root.mainloop()
59

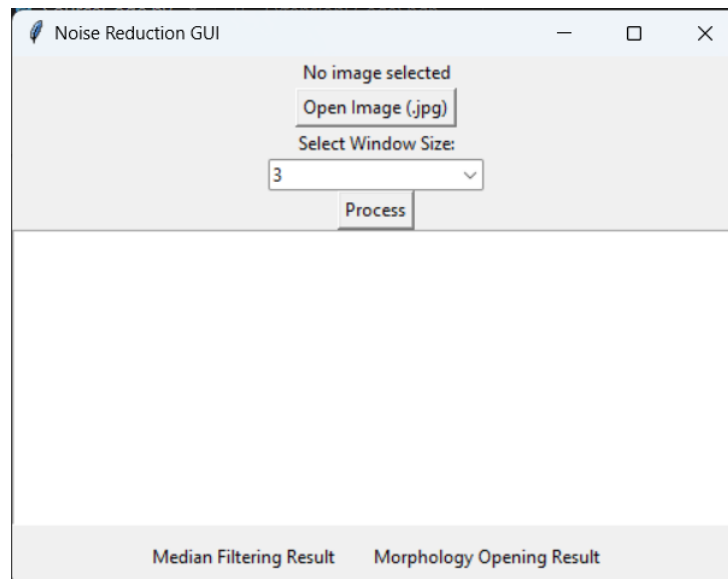
```

Fungsi ini membuka jendela baru yang memperlihatkan semua tahapan hasil proses citra dalam bentuk visual, dari gambar asli hingga hasil setelah filtering.

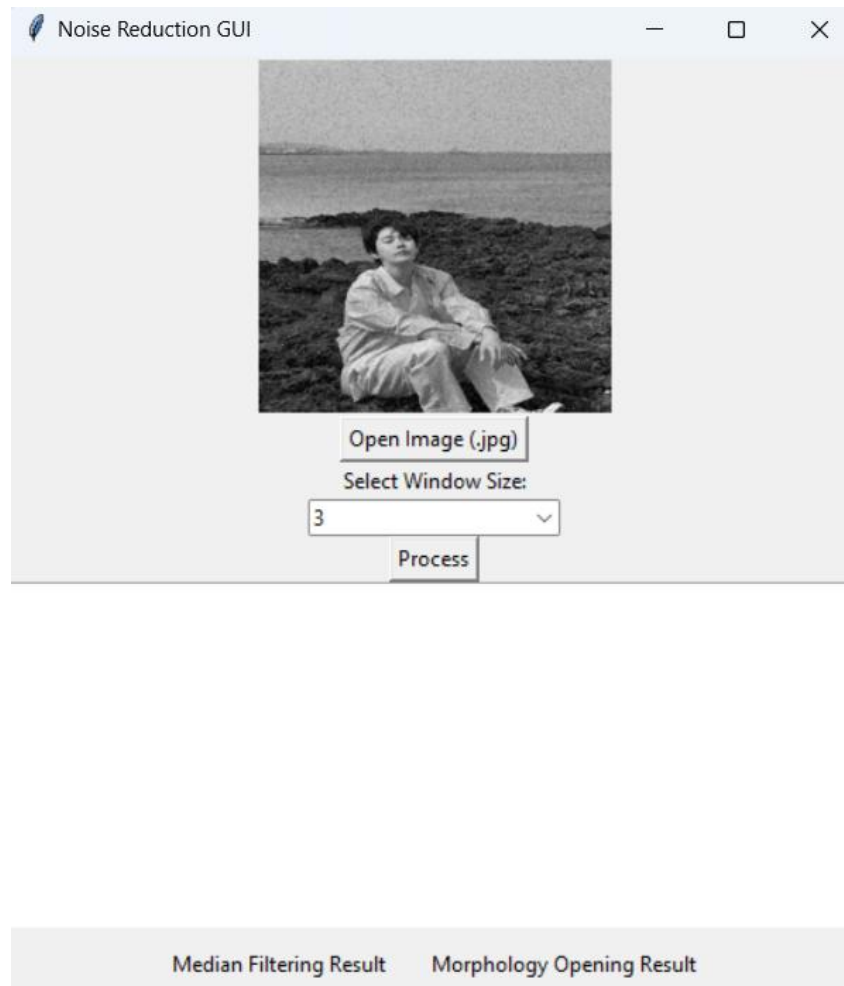
Gambar ditampilkan dalam grid (2 baris x 3 kolom), dan akan otomatis disesuaikan ukurannya saat jendela diubah ukurannya. Hal ini memungkinkan pengguna melihat

perbandingan langsung dan menilai efektivitas dari masing-masing metode reduksi noise.

4.3 Hasil Pengujian



Pengujian dilakukan dengan memanfaatkan antarmuka aplikasi GUI yang dibangun menggunakan Python dan pustaka Tkinter. Tahapan pengujian dimulai dengan pemilihan gambar berekstensi .jpg yang kemudian secara otomatis dikonversi ke dalam format grayscale untuk menyederhanakan representasi warna menjadi 256 skala keabuan.

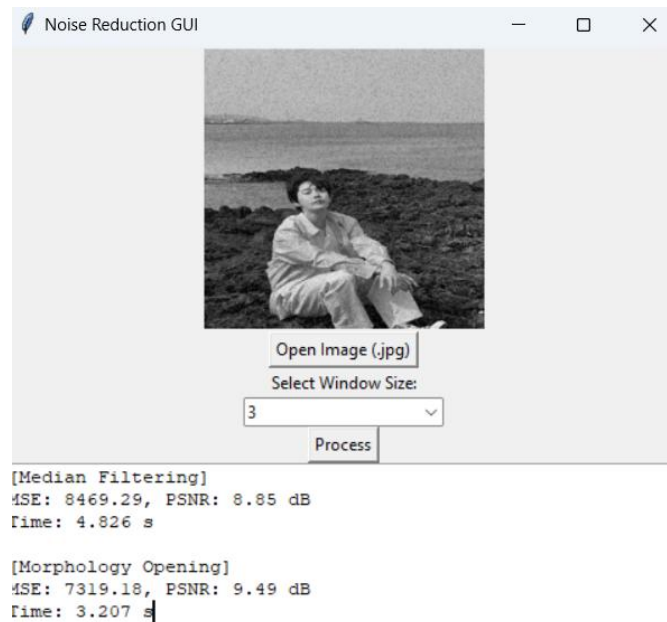


Setelah citra grayscale dihasilkan, dilakukan proses binarisasi otomatis menggunakan metode Otsu Thresholding, yaitu teknik penentuan ambang batas optimal berdasarkan nilai variansi antar kelas dari histogram citra. Proses ini menghasilkan citra biner (hitam-putih) serta informasi nilai ambang (threshold) yang dipilih.

Langkah selanjutnya adalah identifikasi noise, yaitu mendeteksi piksel-piksel bising yang berbeda dari seluruh 8 tetangganya. Piksel-piksel tersebut ditandai dengan warna abu-abu sebagai indikator keberadaan noise pada citra biner.

Setelah noise diidentifikasi, pengguna dipersilakan untuk memilih kernel yang akan digunakan, diantara kernel 3x3, 5x5, atau 7x7, lalu akan diterapkan dua metode reduksi noise yang diuji dan dibandingkan:

- a. Median Filtering
Merupakan metode berbasis statistik lokal, di mana nilai tengah dari jendela (kernel) dipilih untuk menggantikan nilai piksel pusat. Metode ini sangat efektif untuk menangani noise salt-and-pepper.
- b. Morphology Opening
Merupakan gabungan dua operasi morfologi dasar: erosi diikuti dengan dilation. Metode ini menghapus noise kecil (objek terang pada latar belakang gelap) sambil mempertahankan struktur bentuk objek yang lebih besar.



Untuk mengevaluasi performa kedua metode tersebut, digunakan dua metrik kuantitatif:

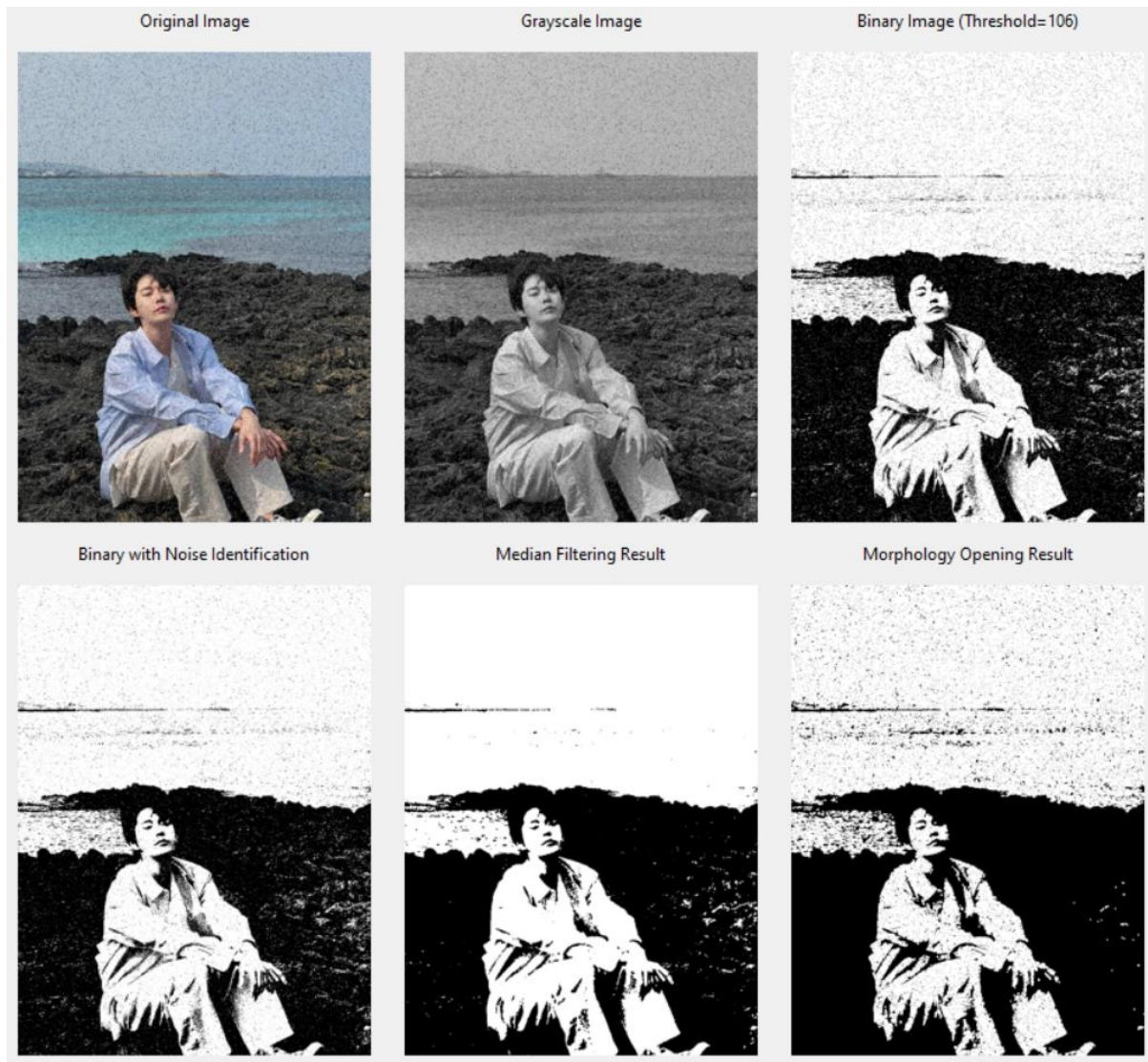
- MSE (Mean Squared Error): Mengukur rata-rata kesalahan kuadrat antara citra hasil filter dan citra referensi asli (grayscale).
- PSNR (Peak Signal-to-Noise Ratio): Mengukur rasio antara sinyal maksimum dan noise, dalam satuan desibel (dB).

Selain itu, waktu eksekusi dari masing-masing metode juga dicatat untuk mengetahui efisiensi komputasi.

Telah dilakukan pengujian dengan memilih kernel 3x3. Dengan didapatkan hasil evaluasi sesuai berikut:

Metode	MSE (Mean Squared Error)	PSNR (dB)	Waktu Eksekusi (s)
Median Filtering	8469,29	8,85	4,826
Morphology Opening	7319,18	9,49	3,207

Visualisasi proses ditampilkan secara langsung dalam jendela baru yang mencakup:



Visualisasi ini sangat membantu untuk melakukan perbandingan visual antar metode, terutama saat nilai MSE dan PSNR tidak menunjukkan perbedaan signifikan.

4.4 Evaluasi dan Pembahasan

Evaluasi hasil pengolahan citra dalam eksperimen ini dilakukan dengan dua pendekatan utama, yaitu evaluasi kuantitatif (numerik) dan evaluasi kualitatif (visual). Tujuan utama dari proses ini adalah untuk menilai efektivitas metode Median Filtering dan Morphological Opening dalam mereduksi noise pada citra biner hasil thresholding.

a. Evaluasi Kuantitatif (Numerik)

Evaluasi kuantitatif dilakukan dengan menghitung dua parameter utama, yaitu Mean Squared Error (MSE) dan Peak Signal-to-Noise Ratio (PSNR). Nilai MSE yang lebih kecil menunjukkan bahwa perbedaan antara citra hasil filter dan citra referensi lebih kecil, sedangkan nilai PSNR yang lebih tinggi menunjukkan kualitas citra yang lebih baik secara objektif.

Adapun hasil pengukuran kuantitatif yang diperoleh dari aplikasi adalah sebagai berikut:

Metode	MSE (Mean Squared Error)	PSNR (dB)	Waktu Eksekusi (s)
Median Filtering	8469,29	8,85	4,826
Morphology Opening	7319,18	9,49	3,207

Berdasarkan data tersebut, metode Morphology Opening memiliki nilai MSE yang lebih kecil (7319.18 dibandingkan 8469.29) dan PSNR yang lebih tinggi (9.49 dB dibandingkan 8.85 dB) dibandingkan metode Median Filtering. Dengan demikian, secara kuantitatif, Morphology Opening dinilai lebih unggul dalam mengurangi noise dari citra biner hasil thresholding.

Selain itu, dari segi waktu eksekusi, Morphology Opening juga menunjukkan waktu proses yang lebih singkat (3.207 detik) dibanding Median Filtering (4.826 detik). Hal ini menjadi nilai tambah dalam konteks efisiensi waktu pengolahan.

b. Evaluasi Kualitatif (Visualisasi)

Evaluasi kualitatif dilakukan dengan membandingkan hasil visual dari masing-masing tahapan pengolahan citra yang ditampilkan pada antarmuka aplikasi. Gambar hasil pengolahan terdiri atas enam bagian yang ditampilkan secara berurutan, yaitu:



Secara visual, hasil dari Morphology Opening menunjukkan reduksi noise yang lebih bersih dan konsisten dibandingkan hasil Median Filtering. Area latar belakang menjadi lebih seragam, dan objek utama (manusia yang duduk di atas batu) tetap terlihat jelas dan tidak mengalami distorsi yang berarti. Sebaliknya, hasil Median Filtering masih menyisakan cukup banyak noise pada latar belakang, terutama pada area laut dan langit.

Selain itu, struktur bentuk objek dalam hasil Morphology Opening tampak lebih terjaga, dengan kontur yang tetap tegas, sedangkan hasil Median Filtering menunjukkan adanya sedikit efek kabur akibat proses penyaringan median, terutama pada tepi objek.

Dalam pengujian ini, pengguna diberikan opsi untuk memilih ukuran kernel yang digunakan, yaitu 3×3 , 5×5 , atau 7×7 . Ukuran kernel mempengaruhi efektivitas proses penyaringan: kernel yang lebih besar cenderung menghilangkan noise lebih banyak, namun berisiko mengaburkan detail objek penting. Pada hasil gambar yang ditampilkan, metode Median Filtering maupun Morphology Opening menggunakan kernel 5×5 sebagai kompromi antara penghilangan noise dan pelestarian detail objek.

Berdasarkan hasil analisis terhadap parameter kuantitatif dan visualisasi kualitatif, metode Morphology Opening menunjukkan performa yang lebih unggul dibandingkan metode Median Filtering dalam proses pereduksian noise pada citra hasil segmentasi.

Dengan mempertimbangkan seluruh aspek tersebut, dapat disimpulkan bahwa metode Morphology Opening merupakan metode yang lebih efektif dan efisien dalam mengurangi noise pada citra biner, sambil tetap menjaga integritas bentuk objek utama.

BAB V

PENUTUP

5.1 Kesimpulan

Berdasarkan hasil implementasi dan pengujian metode Median Filtering dan Morphological pada citra digital yang mengandung Salt & Pepper Noise, dapat disimpulkan beberapa hal berikut:

1. Penelitian ini berhasil mengimplementasikan dua metode reduksi noise pada citra biner menggunakan aplikasi berbasis GUI yang dibangun dengan Bahasa pemrograman Python.
2. Kedua metode, baik Median Filtering maupun Morphological Opening, mampu mereduksi noise secara visual dan numerik. Namun, hasil evaluasi menunjukkan bahwa Morphological Opening pada window 3x3 memberikan performa yang lebih baik, ditinjau dari:
 - Nilai MSE lebih rendah (7319.18) dibandingkan Median Filtering (8469.29)
 - Nilai PSNR lebih tinggi (9.49 dB) dibandingkan Median Filtering (8.85dB)
 - Waktu Proses lebih cepat (3.207 detik) dibandingkan Median Filtering (4.826 detik).
3. Secara visual, hasil pengolahan menggunakan Morphological Opening menghasilkan citra yang lebih bersih dan rapi, dengan objek utama tetap terjaga bentuknya. Sementara itu, Median Filtering masih menyisakan noise dan cenderung mengaburkan detail tepi objek.
4. Dengan demikian, Morphological Opening dinilai lebih efektif dan efisien dalam mereduksi noise pada citra biner dibandingkan Median Filtering, khususnya dalam konteks citra manuskrip hasil digitalisasi.

5.2 Saran

Berdasarkan hasil penelitian dan implementasi yang telah dilakukan, berikut beberapa saran yang dapat di pertimbangkan untuk pengembangan lebih lanjut:

1. Penggunaan Dataset yang lebih beragam
Penelitian selanjutnya disarankan menggunakan lebih dari satu citra dengan variasi noise dan kondisi pencahayaan berbeda untuk menguji konsistensi performa metode.
2. Pengujian dengan jenis noise lain
Selain Salt & Pepper, metode dapat diuji terhadap jenis noise lain seperti Gaussian atau Speckle untuk melihat sejauh mana efektivitas masing-masing metode.
3. Pengembangan GUI yang lebih Interaktif
Antarmuka dapat dikembangkan dengan fitur tambahan seperti zoom, perbandingan sisi-ke-sisi, serta pilihan untuk menyimpan hasil pengolahan.
4. Integrasi dengan teknologi lanjutan
Untuk pengolahan yang lebih kompleks, dapat dikombinasikan dengan teknologi seperti machine learning atau deep learning untuk klasifikasi noise atau otomatisasi pemilihan metode terbaik.
5. Optimasi Kinerja Program
Dapat dilakukan perbaikan efisiensi waktu proses, misalnya dengan menggunakan Pustaka pemrosesan citra yang lebih cepat seperti OpenCV atau memanfaatkan multiprocessing.

DAFTAR PUSTAKA

- Bhutada, S., Yashwanth, N., Dheeraj, P., & Shekar, K. (2022). *Opening and Closing in Morphological Image Processing*. *World Journal of Advanced Research and Reviews*, 15(2), 111–114. Diakses dari <https://wjarr.com/content/opening-and-closing-morphological-image-processing>
- Yasir, A., Satria, W., & Yuanda, P. (2023). *Digital Image Processing Metode Median Filtering dan Morfologi Opening dalam Reduksi Noise Citra*. *Jurnal Warta Dharmawangsa*, 17(4), 1687–1701. Diakses dari <https://jurnal.dharmawangsa.ac.id/index.php/juwarta/article/download/3821/2507>